IMPLEMENTASI ALGORITME ADVANCE ENCRYPTION STANDARD (AES) PADA ENKRIPSI DAN DEKRIPSI QR-CODE

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Dwi Qunita Putri Ambeq Paramarta
NIM: 145150207111121

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI ALGORITME ADVANCED ENCRYPTION STANDARD (AES) PADA ENKRIPSI DAN DEKRIPSI QR-CODE

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Dwi Qunita Putri Ambeq Paramarta
NIM: 145150207111121

Skripsi ini telah diuji dan dinyatakan lulus pada
30 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Ari Kusyanti, S.T, M.Sc	Mahendra Data, S.Kom., M.Kom
NIK: 201102 831228 2 001	NIK: 201503 861117 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 30 Juli 2018

Dwi Qunita Putri Ambeq Paramarta

NIM: 145150207111121

KATA PENGANTAR

Segala puji dan syukur kehadirat Allah SWT, atas limpahan Rahmat dan Karunia-Nya, sehingga saya selaku penulis dapat menyelesaikan skripsi dengan judul "IMPLEMENTASI ALGORITME ADVANCE ENCRYPTION STANDARD (AES) PADA ENKRIPSI DAN DEKRIPSI QR-CODE" sebagai syarat untuk menyelesaikan Program Sarjana (S1) dan memperoleh gelar Sarjana Komputer (S.Kom) pada program Sarjana Fakultas Ilmu Komputer Universitas Brawijaya.

Dalam proses penyusunan skripsi in, saya mengalami berbagai hambatan yang menghadang. Namun semua jambatan-hambatan itu dapat saya lalui berkat adanya dukungan serta bimbingan dari berbagai pihak secara moral dan spiritual. Oleh karena itu, pada kemsempatan ini saya selaku penulis menyampaikan ucapan terima kasih kepada:

1. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku dekan fakultas ilmu komputer universitas brawijaya.
2. Ibu Ari Kusyanti, S.T, M.Sc selaku dosen pembimbing I yang telah membimbing dan memberikan dukungan kepada penulis dalam menyusun skripsi ini.
3. Bapak Mahendra Data, S.Kom., M.Kom selaku dosen pembimbing II yang telah membimbing dan memberikan dukungan kepada penulis dalam menyusun skripsi ini.
4. Kedua orang tua saya, ayah H. Joko Kuncoro, S.Pd, M.M dan Hj. Dra. Ropita selaku mama saya yang telah memberikan dukungan dan doa yang tiada habisnya kepada saya selaku putri keduanya sehingga saya dapat menyelesaikan skripsi ini dengan luar biasa.
5. Kakak saya Bripda Prio Bhudi Setya serta kedua adik saya Ramadhan Siswoyo dan Putri Lina Sabira yang memberikan motivasi dan dukungan dalam menyelesaikan skripsi ini.
6. Teman-teman saya yaitu Risailin Dwi Jaka Fauzi, Yuniar Siska Fatmala, Mahdarani Dwi Laxmi, Kevin Dwiki Saputra, dll. Yang telah memberikan dukungan, semangat dan bantuan kepada saya selaku penulis.

Akhir kata penulis menyadari bahwa dalam penulisan skripsi ini masih jauh dari kesempurnaan. Karena itu, penulis memohon saran dan kritik yang sifatnya membangun demi kesempurnaannya dan semoga bermanfaat bagi kita semua. Amiin.

Malang, 30 Juli 2018

Penulis

dwiqunitaputri@gmail.com

ABSTRAK

Keamanan data merupakan masalah yang sangat penting dalam perkembangan teknologi saat ini. Oleh sebab itu dibutuhkan sebuah cara yang dapat menjaga keamanan merujuk pada perlindungan informasi dari penyingkapan pihak yang tidak sah. Salah satu mekanisme untuk meningkatkan keamanan data adalah dengan menggunakan teknik kriptografi. Ada berbagai macam algoritme dalam kriptografi salah satunya adalah *Algoritme Advance Encryption Standard.* Skripsi ini menggunakan *Algoritme AES* dengam ukuran ekspansi key 128 bit yang akan beroperasi dalam sebuah *array* 4x4. Pada proses *state* enkripsi akan melalui beberapa tahapan yakni *Addroaundkey, Subbyte, Shiftrows,* dan *Mixcolumns* sebanyak 10 kali putaran. Namun pada putaran terakhir tidak dilakukan lagi proses *Mixcolumns* langsung ke proses *Addraoundkey*, dan untuk proses dekripsi merupakan proses kebalikan dari proses enkripsi yakni *InvAddrows, InvShiftrows, InvSubbyte, dan InvMixcolumns* menggunakan kunci *round* yang sama dengan proses enkripsi. AES diimplementasikan dalam bahasa pemrograman PHP dan diterapkan pada QR *Code* karena merupakan sebuah teknologi *labelling* yang dapat menyimpan data dalam bentuk pola yang dapat diisi dengan informasi. Dari hasil implementasi algoritme AES dapat disimpulkan bahwa aplikasi ini dapat mengekripsi semua jenis karakter berupa *string,* huruf, angka, dan simbol. Pada saat mendekripsi QR *Code* aplikasi akan mengaktifkan fungsi kamera dan melakukan *scanning* QR *Code* yang akan menjadi *plaintext* kembali. Waktu eksekusi enkripsi dan dekripsi AES adalah 0.0034 detik untuk proses enkripsi dan untuk proses dekripsi membutuhkan waktu 0.0029 detik.

Kata kunci : *AES,* Enkripsi, Dekripsi, dan *QR Code*

ABSTRACT

Data security are very important in today's technological development. Therefore, it is necessary to find a way to protect the confidentiality and the security from unauthorized accesses. One of the mechanism to increase data security is to use cryptography. There are many form of cryptography, one of them is Advance Encryption Standard Algorithm. This thesis uses AES Algorithm with the size of 128 bit expansion key. That will operate in a 4x4 array. In the state encryption process will go through several stages of Addroaundkey, Subbyte, Shiftrows, and Mixcolumns 10 times round. But in the last round Mixcolumns no longer process directly into the Addraoundkey process, and for the decryption process is a reverse process of the encryption process that InvAddrows, InvShiftrows, InvSubbyte, and InvMixcolumns use the same round key with the encryption process. AES is implemented in PHP programming language and applied to QR-Code because it is a labeling technology that can store data in the form of patterns that can be filled with information. From the results of the implementation of AES Algorithm can be concluded that this application can encrypt all types of characters in the form of strings, alphabet, numbers, and symbols. When decrypting QR-Code the application will activate the camera function and perform QR-Code scanning that will be plaintext again. The execution time of AES encryption and decryption is 0.0034 seconds for the encryption process and for the decryption process takes 0.0029 seconds.

Keywords : AES, Encryption, Decryption, and QR Code.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Privasi dan keamanan merupakan masalah yang sangat penting dalam perkembangan teknologi saat ini. Oleh sebab itu dibutuhkan sebuah cara yang dapat menjaga kerahasiaan dan keamanan merujuk pada perlindungan informasi dari penyingkapan pihak yang tidak sah. Salah satu mekanisme untuk meningkatkan keamanan data adalah dengan menggunakan teknik kriptografi. Ada berbagai macam algoritme dalam kriptografi salah satunya adalah *Algoritme Advance Encryption Standard.* Enkripsi adalah proses yang dilakukan untuk mengubah suatu informasi menjadi serangkaian kode rumit (*ciphertext*) yang sulit diartikan. Dekripsi adalah proses yang mengembalikan informasi yang sudah dienkripsi menjadi sebuah informasi (*plaintext*). (Daemen & Rjmen, 03 September 1999)

Metode di atas bisa diterapkan pada *QR Code,* karena *QR Code* merupakan sebuah teknologi labelling yang dapat menyimpan data dalam bentuk pola yang dapat diisi dengan informasi. *QR Code* merupakan bentuk evolusi dari kode batang dari satu dimensi menjadi dua dimensi yang dikembangkan oleh Denso Wave. Pengenalan pola dilakukan dengan mendeteksi marker atau tanda yang telah diisi dengan informasi yang dibutuhkan. QR merupakan singkatan dari Quick Response. Tujuannya adalah untuk menyampaikan informasi dengan cepat dan mendapatkan respon yang cepat pula. Berbeda dengan kode batang yang hanya menyimpan informasi secara horizontal, *QR Code* mampu menyimpan informasi secara horizontal dan vertikal. Oleh karena itu, QR Code dapat menampung informasi yang lebih banyak misalnya dalam bentuk URL, teks, angka, dll. Karenanya, dengan menggunakan QR Code, kita dapat menyimpan informasi mengenai nomor surat, klasifikasi dokumen, pengirim dan perihal (Denso, 2013). Kelebihan QR Code dibandingkan dengan Barcode adalah: (1) kapasitas atau panjang kata lebih banyak; (2) tipe data yang disimpan pada QR Code beragam dapat berupa angka atau huruf atau gabungan keduanya; (3) QR Code dapat dibaca dari segala arah sehingga kemungkinan gagal dalam membaca sangat kecil; (4) memiliki ketahanan hingga 30%. Sehingga apabila QR Code mengalami kerusakan hingga 30 % dapat tetap terbaca (Denso, 2013). Dengan kelebihan dan manfaat yang dimiliki, QR Code dapat digunakan sebagai sarana identifikasi surat pada saat surat masuk di bagian tata usaha suatu instansi. Otentikasi aman, dicapai dengan menggunakan algoritme penyembunyian data dengan QR *code.*

Pada penelitian yang dilakukan oleh (Sholeh & Muharom, 2016) yang berjudul "SMART PRESENSI MENGGUNAKAN *QR CODE* DENGAN ENKRIPSI VIGENERE" pada penelitian tersebut menggunakan algoritme kriptografi klasik yang ditemukan oleh Giovan Battista Bellaso yaitu algoritme *Vigenere* untuk perlindungan data

atau informasi menggunakan *QR Code*. Namun Pada penelitian yang dilakukan oleh (Harahap, 2016) yang berjudul "ANALISIS PERBANDINGAN ALGORITMA KRIPTOGRAFI KLASIK VIGENERE CIPHER DAN ONE TIME PAD" Yang menyimpulkan bahwa dari segi keamana algoritme *Vigenere* ini memiliki ketahanan yang kurang kuat dari segi kunci. Karena algoritma *Vigenere* menggunakan kunci yang selau berulang sepanjang pesan yang akan dienkripsi. Sehingga memungkinkan terjadinya kecoboran informasi. dan pada penelitian yang dilakukan oleh (Sumandri, 2017) yang berjudul "STUDI MODEL ALGORITMA KRIPTOGRAFI KLASIK DAN MODERN" yang menyimpulkan bahwa terdapat 3 model kriptografi klasik yaitu *Caesar Cipher*, *Vigenere Cipher* dan *Hill Cipher*. Pada penelitian tersebut juga dijelaskan kelemahan dari *Vigenere Cipher* adalah penggunaan Kunci yang berulang-ulang. Sementara kriptografi modern menggunakan mode bit biner (0 dan 1) yang di bentuk dari kode ASCII, sehingga mempunyai tingkat kesulitan yang kompleks. Kekuatan kriptografi modern ada pada kuncinya (*key*). Kunci yang digunakan pada kriptografi modern terdiri dari tiga jenis yakni: simetris; asimetris; dan hibrida. Contoh kriptografi modern yaitu MD5, RC4, AES dan lain-lain.

Dari hasil penelitian diatas dapat di tarik kesimpulan bahwa algoritme *vigenere* memiliki ketahanan yang kurang kuat dalam pembuatan kunci. Metode yang digunakan algoritme *vigenere* dalam pembuatan kunci yaitu diambil dari substitusi polyalphabetic di mana setiap alfabet bisa diganti dengan beberapa huruf cipher. Sedangkan AES memiliki tingkat keamanan yang lebih baik dibandingkan dengan algoritme yang ada sebelumnya baik dari segi kunci maupun ukuran blok, jadi AES memiliki ketahanan yang lebih baik dalam mengamankan data jika dibandingkan dengan algoritme viginere. Berdasarkan alasan yang telah dipaparkan di atas, pada penelitian ini membahas tentang implementasi algoritme *Advanced Encryption Standard* (AES) untuk enkripsi dan dekripsi pada QC *code.*

1.2 Rumusan masalah

Dengan latar belakang penelitian yang telah disebutkan sebelumya, maka rumusan masalah dari penulisan skripsi ini dapat diuraikan sebagai berikut:

1. Bagaimana implementasi AES untuk enkripsi dan dekripsi pada QR *code*?
2. Bagaimana Melakukan validasi *plaintext* dan *ciphertext* hasil enkripsi dan dekripsi dengan *test vector* algoritme aes?
3. Bagaimana kinerja pemrosesan enkripsi dan dekripsi algoritme AES pada *QR Code*?

1.3 Tujuan

Tujuan dari penelitian ini dengan dapat diuraikan sebagai berikut :

1. Mengimplementasikan algoritme *Advanced Encryption Standard* (AES) untuk enkripsi dan dekripsi pada QR *Code.*

2. Melakukan validasi *plaintext* dan *ciphertext* hasil enkripsi dan dekripsi dengan *test vector* algoritme aes.
3. Mengetahui kinerja pemrosesan enkripsi dan dekripsi algoritme AES pada *QR Code*

1.4 Manfaat

Manfaaat yang dapat diperoleh dari penelitian ini adalah :

1. Pengguna dapat meningkatkan keamanan dan kerahasiaan dengan mengimplementasikan algoritme *Advanced Encryption Standard* (AES) untuk enkripsi dan dekripsi pada QR *Code*.
2. Pengguna dapat mengetahui kinerja pemrosesan enkripsi dan dekripsi algoritme AES pada QR *Code.*
3. Pengguna dapat menjaga *privacy* / *confidentiality* informasi untuk menghindari upaya penyadapan, pembajakan, dan hal yang menyebabkan kebocoran dan manipulasi informasi melalui teknik enkripsi dan dekripsi.

1.5 Batasan masalah

Sesuai dengan rumusan masalah diatas, maka batasan masalah dalam penelitian ini dibatasi oleh hal-hal sebagai berikut:

1. Penelitian berfokus dengan implementasi algoritme *Advanced Encryption Standard* (AES) untuk enkripsi dan dekripsi pada QR *Code.*
2. Proses penelitian yang dilakukan mencakup enkripsi dan dekripsi terhadap QR Code.
3. Algoritme AES yang digunakan adalah AES tipe 128 bit dengan key max 16 karakter.
4. Kemampuan Scanning pada perangkat (leptop) yang digunakan hanya mampu menscan sebanyak 60 kata dikarenakan kualitas kamera pada perangkat.

1.6 Sistematika pembahasan

Sistematika pembahasan memberikan gambaran dan uraian dari penyusunan skripsi secara garis besar yang meliputi beberapa bab, antara lain:

BAB I PENDAHULUAN

Berisi tentang latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah dalam pengambilan topik skripsi tentang implementasi algoritme *Advanced Encryption Standard* (AES) untuk enkripsi dan dekripsi pada QC *code*, dan sistematika penulisan dari skripsi ini.

BAB II TINJAUAN PUSTAKA

Berisi kajian pustaka, dan sumber-sumber yang berhubungan dengan permasalahan dalam skripsi antara lain mengenai

kriptografi, algoritme AES serta teori-teori lainnya sebagai dasar penulisan skripsi.

BAB III METODOLOGI PENELITIAN

Bab ini menjelaskan langkah-langkah penelitian serta metode yang akan dipakai dalam penelitian ini. Langkah-langkah yang dimaksud antara lain, Studi literatur, fungsi algoritme aes perancangan sistem AES, implementasi dan pengujian dari sistem AES yang telah di buat.

BAB IV PERANCANGAN

Pada bab ini dijelaskan analisis dan perancangan implementasi algoritme *Advanced Encryption Standard* (AES) pada enkripsi dan dekripsi *QR Code* yang dapat menjawab permasalahan yang telah diuraikan pada rumusan masalah.

BAB V IMPLEMENTASI DAN PENGUJIAN

Memuat tentang implementasi algoritma aes serta hasil pengujiannya berdasarkan perancangan yang telah dibuat sebelumnya.

BAB VI PENUTUP

Bab ini membahas kesimpulan yang didapat dari perancangan, pengujian sistem enkripsi dan dekripsi menggunakan AES, analisis aplikasi yang dikembangkan dalam penelitian ini dan juga berisi saran-saran yang berguna untuk penelitian selanjutnya yang menggunakan penelitian ini sebagai referensi.

BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini terdapat dasar-dasar teori yang mendukung penelitian ini. Dasar-dasar teori yang dimaksud adalah sistem keamanan, algoritme AES, enkripsi dan dekripsi, ilustrasi enkripsi dan dekripsi AES, bahasa PHP, dan *QR Code.*

2.1 Kajian Pustaka

Dalam penelitian ini, akan membahas tentang penerapan algoritma AES pada *QR Code*. Algoritme *Advanced Encryption Standard* (AES) merupakan teknik atau algoritme kriptografi penyandian pesan yang menggunakan teknik *blok simetris*. (Daemen & Rjmen, 03 September 1999) algoritme AES ini merupakan algoritme yang modern yang mana lebih baik dari pada algoritme klasik. Menurut penelitian yang dilakukan oleh (Sumandri, 2017) yang berjudul "STUDI MODEL ALGORITMA KRIPTOGRAFI KLASIK DAN MODERN" yang menyimpulkan bahwa terdapat 3 model kriptografi klasik yaitu *Caesar Cipher*, *Vigenere Cipher* dan *Hill Cipher.* Pada penelitian tersebut juga dijelaskan kelemahan dari *Vigenere Cipher* adalah penggunaan Kunci yang berulang-ulang. Sementara kriptografi modern menggunakan mode bit biner (0 dan 1) yang di bentuk dari kode ASCII, sehingga mempunyai tingkat kesulitan yang kompleks. Kekuatan kriptografi modern ada pada kuncinya (*key*). Kunci yang digunakan pada kriptografi modern terdiri dari tiga jenis yakni: *simetris*, *asimetris*, dan *hibrida.* Contoh kriptografi modern yaitu *MD5, RC4, AES* dan lain-lain.

Pada penelitian yang dilakukan oleh (Harahap, 2016) yang berjudul "ANALISIS PERBANDINGAN ALGORITMA KRIPTOGRAFI KLASIK VIGENERE CIPHER DAN ONE TIME PAD" Yang menyimpulkan bahwa dari segi keamana algoritme *Vigenere* ini memiliki ketahanan yang kurang kuat dari segi kunci. Karena algoritma *Vigenere* menggunakan kunci yang selau berulang sepanjang pesan yang akan dienkripsi.

Oleh karena itu pada penelitian ini menerapkan algoritme AES karena termasuk dalam kriptografi modern. Dan *QR Code* sebagai media penelitiannya karena *QR Code* merupakan sebuah teknologi *labelling* yang dapat menyimpan data dalam bentuk pola yang dapat diisi dengan informasi. Pengenalan pola dilakukan dengan mendeteksi marker atau tanda yang telah diisi dengan informasi yang dibutuhkan. Tujuannya adalah untuk menyampaikan informasi dengan cepat dan mendapatkan respon yang cepat pula. (Denso, 2013)

2.2 Sistem Keamanan

Sistem keamanan adalah sistem yang berguna untuk mengatasi segala bentuk penyebab kerugian dalam bentuk fisik maupun non fisik. sistem keamanan melingkupi lima aspek, meliputi (Widiyanto, 2007) :

1. *Privacy / Confidentiality*
   Aspek *privacy* atau *confidentiality* adalah sebuah tindakan yang dilakukan untuk menjaga informasi dari orang yang tidak berhak mengakses informasi tersebut. *Privacy* lebih ke arah data-data yang bersifat rahasia sedangkan *confidentiality* berhubungan dengan data yang diberikan kepada pihak lain dengan maksud dan tujuan tertentu.
2. *Integrity*
   Aspek *integrity* atau integritas lebih menekankan bahwa suatu informasi tidak boleh diubah tanpa adanya izin dari pemilik informasi tersebut. Jika terdapat perbedaan maka boleh dibilang aspek integritas tidak tercapai. Adanya virus, *trojan horse*, dan sejenisnya merupakan salah satu hal yang umumnya mengakibatkan pengubahan sebuah informasi.
3. *Authentication*
   Aspek ini berhubungan dengan metode untuk menyatakan bahwa informasi betul-betul asli, orang yang mengakses atau memberikan informasi adalah betul-betul orang yang dimaksud, atau server yang ditujukan adalah server yang asli.
4. *Availability*
   Aspek *availability* berhubungan dengan ketersediaan sebuah data atau informasi. Data maupun informasi tersebut hanya dapat digunakan oleh yang berhak.
5. *Access Control*
   Aspek access control berhubungan dengan cara pengaturan akses kepada informasi. Misalnya, seorang administrator memiliki hak akses penuh terhadap sebuah komputer, tetapi hal ini tidak berlaku bagi *account guest* ataupun *limited account* lainnya.

2.3 Algoritme AES

Advanced Encryption Standard (AES) merupakan teknik atau algoritme kriptografi penyandian pesan yang menggunakan teknik *block simetris*. Algoritme ini dikembangkan oleh dua kriptografer yang berasal dari Belgia, yaitu Dr. Joan Daemen dan Dr. Vincent Rijmen pada tahun 1997. Mereka berdua mengajukan algoritme ini sebagai proposal Rijndael bagi AES, dan pada November 2001 disahkan sebagai proposal terpilih bagi AES oleh *National Institute of Standard and Technology* (NIST) (Daemen & Rjmen, 03 September 1999).

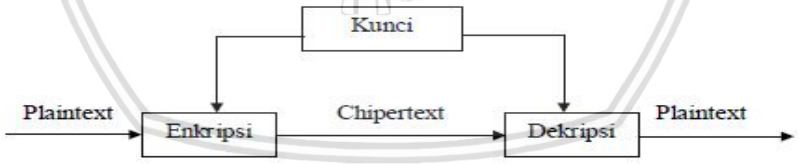
AES sendiri memiliki tipe yang terbagi berdasarkan panjang *block* data seperti AES-128, AES-192, AES-256 dimana masing-masing AES memiliki panjang blok sebanyak 128 bit, 192 bit, dan 256 bit. Perbedaan panjang *block* tersebut berakibat berbedanya jumlah perputaran yang harus dilakukan pada masing-masing panjang kunci perbedaan tersebut dapat dilihat pada table 2.1.

Tabel 2.1 Tabel perbandingan jumlah putaran pada AES

Jenis	Jumlah Putaran (Nr)
AES-128	10
AES-192	12
AES-256	14

2.4 Enkripsi dan Dekripsi

Menurut (Ariyus, 2006), enkripsi merupakan hal yang sangat penting dalam kriptografi untuk mengamankan sebuah informasi agar pesan yang dikirimkan terjaga kerahasiaannya. Informasi (yang disebut *plaintext*) diubah menjadi serangkaian kode rumit yang sulit diartikan (*ciphertext*). Enkripsi sendiri bisa diartikan sebagai *cipher* atau kode. Berdasarkan ISO 7498-2, terminologi yang lebih tepat digunakan untuk menamakan proses ini adalah "*encipher*". Sedangkan dekripsi merupakan kebalikan dari proses enkripsi. Dekripsi yaitu proses mengubah kembali pesan yang telah dienkripsi menjadi pesan aslinya, yang disebut dengan dekripsi pesan. Berdasarkan ISO 7498-2, terminologi yang lebih tepat untuk menamakan proses ini adalah "*decipher*". Dibutuhkan kunci untuk mengubah *plaintext* menjadi *ciphertext*, begitu juga sebaliknya. Tanpa kunci, *plaintext* tidak bisa melakukan enkripsi pesan menjadi *ciphertext*, juga sebaliknya. Kerahasiaan kunci ini sangatlah penting, apabila kerahasiaannya terbongkar maka isi pesan akan terbongkar. Berikut adalah skema ilustrasi proses enkripsi dan dekripsi.



Gambar 2.1 Skema proses enkripsi dan dekripsi

Sumber: Munir (2006)

Pada gambar 2.1 mengilustrasikan sebuah pesan atau plaintext dienkripsi menggunkan kunci sehingga menjadi ciphertext yang akan didekripsi menggunakan kunci untuk menghasilkan plaintext kembali sehingga dapat dibaca.

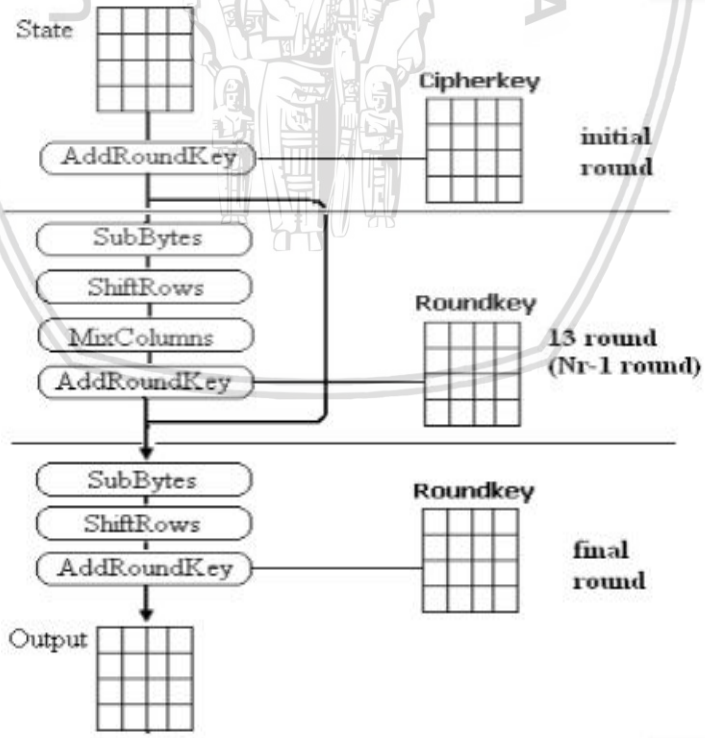
2.5 Ilustrasi Enkripsi dan Dekripsi AES

Proses enkripsi dan dekripsi algoritme AES terdiri dari 4 jenis transformasi bytes, yaitu SubBytes, ShiftRows, Mixcolumns, dan AddRoundKey. Pada awal proses enkripsi, input yang telah dicopykan ke dalam state akan mengalami

transformasi byte AddRoundKey. Setelah itu, state akan mengalami transformasi SubBytes, ShiftRows, MixColumns, dan AddRoundKey secara berulang-ulang sebanyak Nr. Proses ini dalam algoritme AES disebut sebagai round function. Round yang terakhir agak berbeda dengan round-round sebelumnya dimana pada round terakhir, state tidak mengalami transformasi MixColumns. Ilustrasi proses enkripsi AES dapat digambarkan seperti pada Gambar 2.4. Transformasi cipher dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *invers* cipher yang mudah dipahami untuk algoritme AES. Transformasi byte yang digunakan pada *invers* cipher adalah *InvShiftRows, InvSubBytes, InvMixColumns, dan AddRoundKey.*

2.5.1 Ilustrasi Enkripsi AES

Pada tabel 2.2 dijabarkan bahwa AES memiliki ukuran *block* yang tetap sepanjang 128 bit dan ukuran kunci sepanjang 128, 192, atau 256 bit. Menurut (Munir, 2006) berdasarkan ukuran *block* yang tetap, AES bekerja pada matriks berukuran 4x4 di mana tiap-tiap sel matriks terdiri atas 1 byte (8 bit). *Blokcipher* tersebut dalam pembahasan ini akan diasumsikan sebagai sebuah kotak. Setiap *plaintext* akan dikonversikan terlebih dahulu ke dalam blok-blok tersebut dalam bentuk heksadesimal. Barulah kemudian blok itu akan diproses dengan metode yang akan dijelaskan. Alur enkripsi dapat dilihat pada gambar 2.2 berikut ini:



Gambar 2.2 Ilustrasi enkripsi AES

Sumber: Munir (2006)

1. *AddroundKey*

   Pada proses enkripsi dan dekripsi AES proses *AddRoundKey* sama, sebuah *round key* ditambahkan pada state dengan operasi XOR. Setiap *round key* terdiri dari Nb word dimana tiap *word* tersebut akan dijumlahkan dengan word atau kolom yang bersesuaian dari *state* sehingga : [s c ,s c ,s c ,s c] [s ,0 c ,s ,1 c ,s ,2 c ,s ,3 c] [wround*Nb c]untuk 0 c Nb ' ,3 ' ,2 ' ,1 ' ,0 = ⊕ + ≤ ≤ [wi] adalah *word* dari *key* yang bersesuaian dimana i = *round**Nb+c. Transformasi *AddRoundKey* pada proses enkripsi pertama kali pada round = 0 untuk *round* selanjutnya *round* = *round* + 1, pada proses dekripsi pertama kali pada *round* = 14 untuk *round* selanjutnya *round* = *round* - 1.

2. *SubByte*

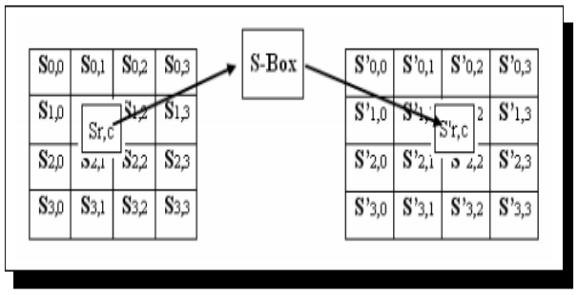
   SubBytes merupakan transformasi *byte* dimana setiap elemen pada *state* akan dipetakan dengan menggunakan sebuah tabel substitusi (S-Box). Tabel substitusi S-Box akan dipaparkan dalam gambar 2.3 berikut :

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Gambar 2.3 Tabel S-Box *SubBytes*

Sumber: Munir (2006)

Untuk setiap *byte* pada *array state*, misalkan S[r, c] = xy, yang dalam hal ini xy adalah digit *heksadesimal* dari nilai S[r, c], maka nilai substitusinya, dinyatakan dengan S'[r, c], adalah elemen di dalam tabel subtitusi yang merupakan perpotongan baris x dengan kolom y. Gambar 2.4 mengilustrasikan pengaruh pemetaan *byte* pada setiap *byte* dalam *state.*

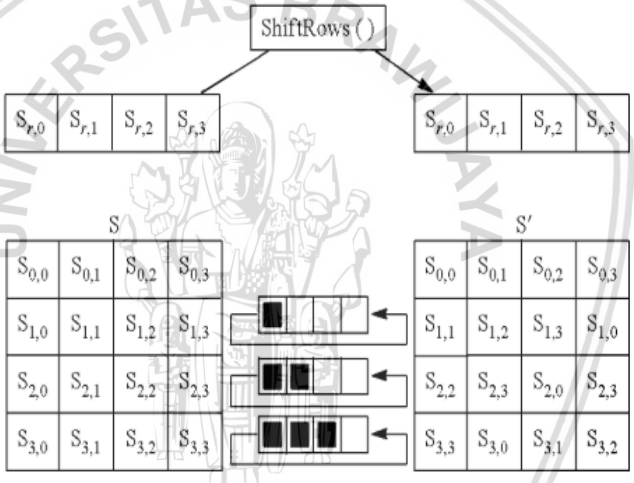


Gambar 2.4 Ilustrasi *SubByte*

Sumber: Munir (2006)

3. *ShiftRows*

Transformasi *Shiftrows* pada dasarnya adalah proses pergeseran bit dimana bit paling kiri akan dipindahkan menjadi bit paling kanan (rotasi bit). Proses pergeseran *Shiftrow* ditunjukkan dalam Gambar 2.5 berikut:



Gambar 2.5 Ilustrasi *ShiftRows*

Sumber: Munir (2006)

4. *MixColumns*

MixColumns mengoperasikan setiap elemen yang berada dalam satu kolom pada *state*. Secara lebih jelas, transformasi *mixcolumns* dapat dilihat pada perkalian matriks berikut ini:

$$\begin{bmatrix} S'_{0,C} \\ S'_{1,C} \\ S'_{2,C} \\ S'_{3,C} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,C} \\ S_{1,C} \\ S_{2,C} \\ S_{3,C} \end{bmatrix}$$

Gambar 2.6 Transformasi *Mixcolumns*

Sumber: Munir (2006)

Hasil dari perkalian matriks di atas dapat dianggap seperti perkalian yang ada di bawah ini :

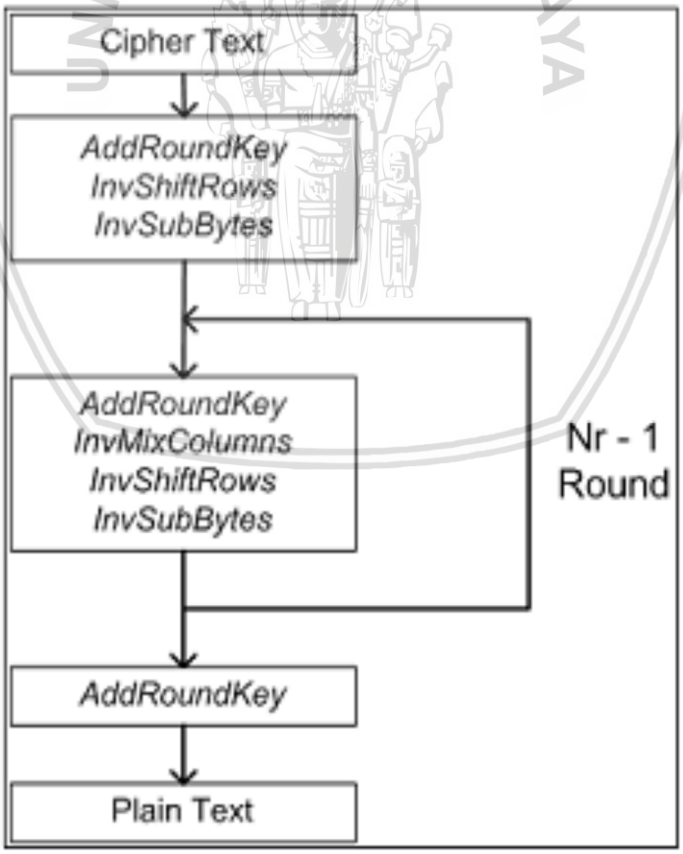
$$s'_{0,c} = (\{02\} \bullet s_{o,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$
$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$
$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$
$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c})$$

Gambar 2.7 Hasil perkalian matriks *Mixcolumns*

Sumber: Munir (2006)

2.5.2 Ilustrasi Dekripsi AES

Menutrut (Munir, 2006) transformasi *cipher* dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *invers cipher* yang mudah dipahami untuk algoritme AES. Transformasi *byte* yang digunakan pada *invers cipher* adalah *InvShiftRows, InvSubBytes, InvMixColumns,* dan *AddRoundKey*. Alur dekripsi dapat dilihat pada gambar 2.8 berikut ini :

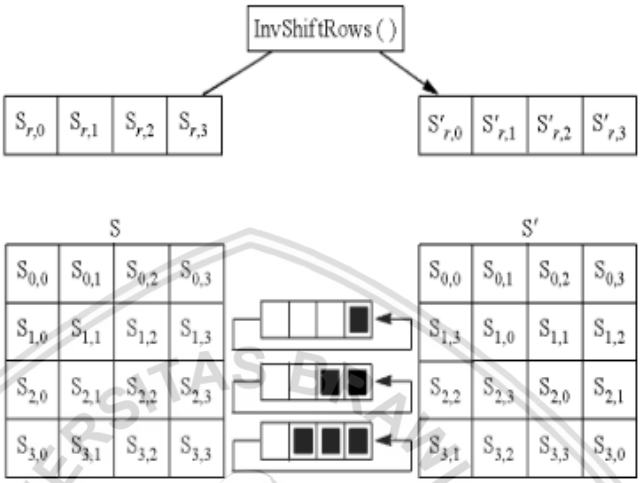


Gambar 2.8 Ilustrasi dekripsi pada AES

Sumber: Munir (2006)

1. *InvShiftRows*

   InvShiftRows adalah transformasi *byte* yang berkebalikan dengan transformasi *ShiftRows.* Pada transformasi *InvShiftRows*, dilakukan pergeseran bit ke kanan sedangkan pada *ShiftRows* dilakukan pergeseran bit ke kiri. Ilustrasi transformasi *InvShiftRows* terdapat pada Gambar 2.9:



Gambar 2.9 Transformasi *InvShiftRows*

Sumber: Munir (2006)

2. *InvSubBytes*

   InvSubBytes juga merupakan transformasi *bytes* yang berkebalikan dengan transformasi *SubBytes*. Pada *InvSubBytes*, tiap elemen pada *state* dipetakan dengan menggunakan tabel *Invers* S-Box. Tabel *Invers* S-Box akan ditunjukkan dalam gambar 2.10 berikut:

	y															
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

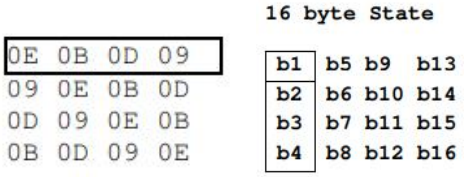
(x)

Gambar 2.10 Tabel *InvSubBytes*

Sumber: Munir (2006)

3. *InvMixColumns*

Setiap kolom dalam state dikalikan dengan matriks perkalian dalam AES. Perkalian dalam matriks dapat dituliskan :



Gambar 2.11 Multiplication Matriks

Sumber: Barent (2014)

Hasil dari perkalian dalam matriks adalah :

Gambar 2.12 Hasil perkalian kolom dalam state dengan matriks

Sumber: Barent (2014)

2.6 Bahasa Pemograman PHP

Menurut (Haryana, 2008), *Hypertext Prepocessor* (PHP) merupakan salah satu bahasa pemograman berbasis web dimana sistem yang diterapkan adalah pada sisi server side. PHP dapat disisipkan diantara *skrip-skrip* bahasa HTML dan arena bahasa *server side* lainnya, dengan itu maka PHP akan dieksekusi secara langsung pada *server*. Sedangkan *browser* akan mengeksekusi halaman web tersebut melalui *server* yang kemudian akan menerima tampil hasil dalam bentuk HTML, sedangkan kode PHP itu sendiri tidak akan dapat terlihat. Kelebihan PHP yaitu:

a. Web menggunakan PHP dapat dengan mudah dibuat dan memiliki kecepatan akses yang cukup tinggi.
b. Skrip-skrip PHP dapat berjalan dalam *web server* yang berbeda dan dalam sistemoperasi yang berbeda.
c. PHP dapat berjalan disistem operasi UNIX, windows dan macintosh.
d. PHP diterbitkan secara gratis.
e. PHP juga dapat berjalan pada web server Microsoft Personal Web Server, Apache, IIS, Xitami dan sebagainya.

f. PHP adalah termasuk bahasa embedded (bisa ditempel atau diletakan dalam tag HTML)
g. PHP termasuk *server side* programming PHP juga mendukung komunikasi dengan layanan lain melalui protokol IMAP, SNMP, NNTP, POP3 dan HTTP.

Fungsi-fungsi yang ada di PHP tidak *case* sensitive tetapi variabelnya *case* sensitive (membedakan hurup besar dan kecil). Kode PHP diawali dengan tanda lebih kecil (). Konsep kerja HTML diawali dengan permintaan suatu halaman web oleh browser. Berdasarkan URL (*Uniform Resource Locator*) atau dikenal dengan internet, browser mendapat alamat dari *web server*, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh web *server*. Selanjutnya, *web server* akan mencarikan berkas yang diminta dan membrikan isinya ke *browser*. *Browser* yang mendapatkan isinya segera melakukan proses penerjemahan kode HTML dan menampilkannya ke layar pemakai (klien). Pada PHP prinsip kerjanya sama, hanya saja ketika berkas PHP yang diminta didapatkan oleh *web server*, isinya segera dikirimkan ke mesin PHP dan mesin inilah yang memproses dan memberikan hasilnya (berupa kode HTML) ke *web server*. Selanjutnya, *web server* menyampaikannya ke klien.

2.7 Quic Response Code (CR-Code)

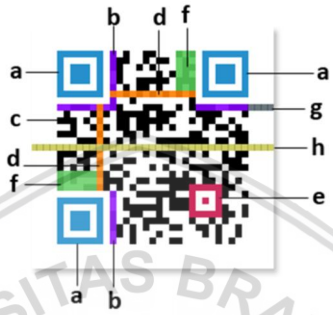
Quick Response Code sering di sebut *QR Code* atau Kode *QR* adalah semacam simbol dua dimensi yang dikembangkan oleh Denso Wave yang merupakan anak perusahaan dari Toyota sebuah perusahaan Jepang pada tahun 1994. Tujuan dari *QR Code* ini adalah untuk menyampaikan informasi secara cepat dan juga mendapat tanggapan secara cepat. Pada awalnya *QR Code* digunakan untuk pelacakan bagian kendaraan untuk *manufacturing*. Namun sekarang, telah digunakan untuk komersil yang ditujukan pada pengguna telepon seluler. *QR Code* adalah perkembangan dari *barcode* atau kode batang yang hanya mampu menyimpan informasi secara horizontal sedangkan *QR Code* mampu menyimpan informasi lebih banyak, baik secara horizontal maupun vertikal.

Gambar 2.13 Contoh *QR Code*

Sumber: Denso (2011)

QR Code biasanya berbentuk persegi putih kecil dengan bentuk geometris hitam dapat dilihat di gambar 2.13, meskipun sekarang banyak yang telah berwarna dan digunakan sebagai *brand* produk. Informasi yang dikodekan dalam *QR Code* dapat berupa *URL*, nomor telepon, pesan *SMS*, *V-Card*, atau teks apapun (Ashford, 2010). *QR Code* telah mendapatkan standarisasi internasional ISO/IEC18004 dan Jepang JIS-X-0510 (Denso, 2011).

2.7.1 Anatomi *QR Code*



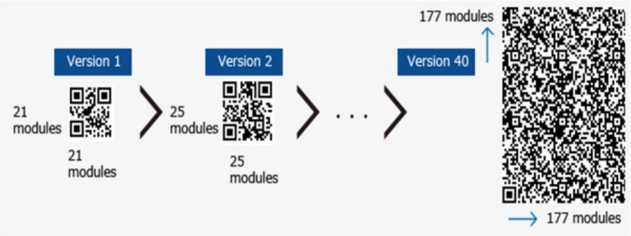
Gambar 2.14 Anatomi *QR Code*

Sumber: Ariadi (2011)

Beberapa penjelasan anatomi *QR Code* Menurut (Ariadi, 2011) antara lain :

a. *Finder Pattern* berfungsi untuk identifikasi letak *QR Code*.
b. *Format Information* berfungsi untuk informasi tentang *error correction level* dan *mask pattern*.
c. *Data* berfungsi untuk menyimpan data yang dikodekan.
d. *Timing Pattern* merupakan pola yang berfungsi untuk identifikasi koordinat pusat *QR Code*, berbentuk modul hitam putih.
e. *Alignment Pattern* merupakan pola yang berfungsi memperbaiki penyimpangan *QR Code* terutama distorsi non linier.
f. *Version Information* adalah versi dari sebuah *QR Code*.
g. *Quiet Zone* merupakan daerah kosong di bagian terluar *QR Code* yang mempermudah mengenali pengenalan *QR* oleh sensor *CCD*.
h. *QR Code version* adalah versi dari *QR Code* yang digunakan.

2.7.2 Versi *QR Code*



Gambar 2.15 Versi *QR Code*

Sumber: Denso (2011)

QR Code dapat menghasilkan 40 versi yang berbeda dari versi 1 (21 x 21 modul) sampai versi 40 (177 x 177 modul). Tingkatan Versi *QR Code* 1 dan 2 berbeda 4 modul berlaku sampai dengan versi 40. Setiap versi memiliki konfigurasi atau jumlah modul yang berbeda. Modul ini mengacu pada titik hitam dan putih yang membentuk suatu *QR Code*. Setiap versi *QR Code* memiliki kapasitas maksimum data, jenis karakter dan tingkat koreksi kesalahan. Jika Jumlah data yang ditampung banyak maka modul yang yang akan diperlukan dan menjadikan *QR Code* menjadi lebih besar (Denso, 2011).

2.7.3 Mengoreksi Kesalahan *QR Code*

QR Code mampu mengoreksi kesalahan dan pengembalian data dalam pembacaan kode apabila *QR Code* kotor atau rusak. Menurut (Denso, 2011), Ada 4 tingkatan koreksi kesalahan dalam *QR Code*:

Tabel 2.2 level koreksi

Level Koreksi kesalahan Jumlah	Jumlah Perkiraan Koreksi
L	7%
M	15%
Q	25%
H	30%

Sumber: Denso (2011)

Semakin tinggi tingkat koreksi kesalahan semakin besar juga versi *QR Code*. Faktor lokasi dan lingkungan operasi perlu di timbangkan dalam menentukan level *QR Code*. Level Q dan H baik digunakan di pabrik yang kotor, sedangkan L untuk tempat yang bersih. Level yang sering digunakan adalah level M dengan perkiraan koreksi mencapai 15%.

2.7.4 Manfaat *QR Code*

Beberapa manfaat yang terdapat pada *QR Code* menurut (Denso, 2011)) antara lain:

1. Kapasitas tinggi dalam menyimpan data sebuah *QR Code* tunggal dapat menyimpan sampai 7.089 angka.
2. Ukuran yang kecil sebuah *QR Code* dapat menyimpan jumlah data yang sama dengan *barcode 1D* dan tidak memerlukan ruang besar.
3. Dapat mengoreksi kesalahan tergantung pada tingkat koreksi kesalahan yang dipilih, data pada *QR code* yang kotor atau rusak sampai 30% dapat diterjemahkan dengan baik.
4. Banyak jenis data *QR Code* dapat menangani angka, abjad, simbol, karakter bahasa Jepang, Cina atau Korea dan data biner.

5. Kompensasi distorsi *QR Code* tetap dapat dibaca pada permukaan melengkung atau terdistorsi.
6. Kemampuan menghubungkan

Sebuah *QR Code* dapat dibagi hingga 16 simbol yang lebih kecil agar sesuai dengan ruang. Simbol-simbol kecil yang dibaca sebagai kode tunggal apabila di *scan* menurut urutan.

2.7.5 Macam-macam *QR Code*

1. *QR Code* model 1 dan model 2

Gambar 2.16 *QR Code* Model 1

Sumber: Desno (2013)

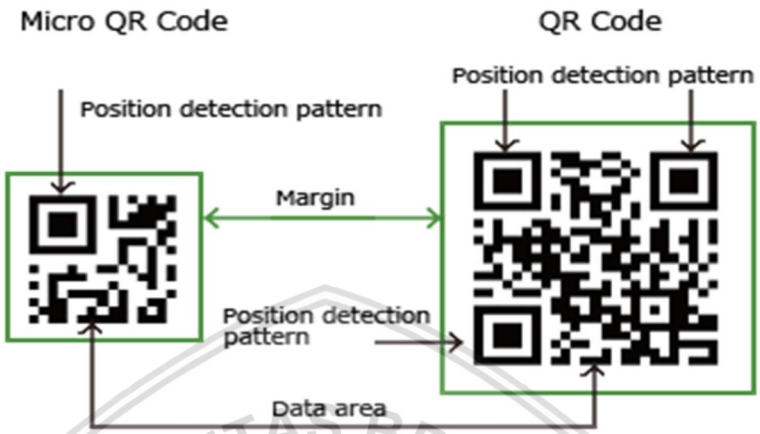
Model 1 adalah *QR Code* asli, dapat menampung 1.167 angka dengan versi maksimum 14 (73 x 73 modul) (Denso, 2013).

Gambar 2.17 *QR Code* Model 2

Sumber: Denso (2013)

Model 2 adalah penyempurnaan dari model 1 dengan versi terbesar 40 (177 x 177 modules), yang mampu menyimpan sampai 7.089 angka (Denso, 2013).

2. Micro *QR Code*

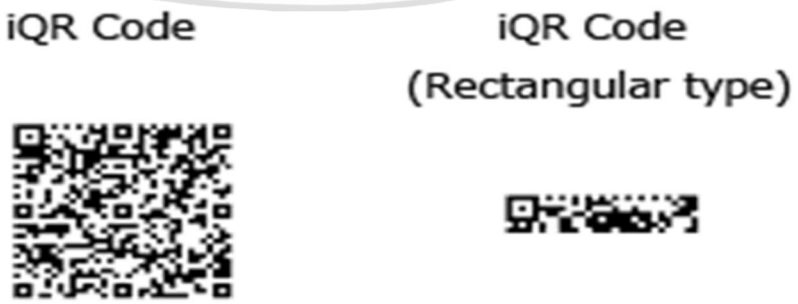


Gambar 2.18 Micro *QR Code*

Sumber: Denso (2013)

Versi terbesar dari kode ini adalah M4 (17 x 17 modul) yang dapat menyimpan hingga 35 angka. Fitur utama dari *Micro QR Code* adalah hanya memiliki satu pola deteksi posisi, dibandingkan dengan regular *QR Code* yang memerlukan sejumlah tempat karena pola deteksi posisi yang terletak di tiga sudut simbol. *QR Code* biasa membutuhkan setidaknya empat modul yang lebar di sekitar simbol, sedangkan *Micro QR Code* hanya membutuhkan cukup dua modul margin. Konfigurasi *Micro QR Code* memungkinkan pencetakan di tempat lebih kecil dari *QR Code* (Denso, 2013).

3. i*QR Code*



Gambar 2.19 i*QR Code*

Sumber: Denso (2013)

Kode yang dapat dihasilkan dari salah satu modul, persegi atau persegi panjang. Dan dapat di cetak sebagai kode versi hitam putih atau kode pola *dot* (bagian penanda). Versi terbesar dari kode ini dapat mencapai 61 (422x 422 modul), yang dapat menyimpan 40.000 angka (Denso, 2013).

4. LogoQ

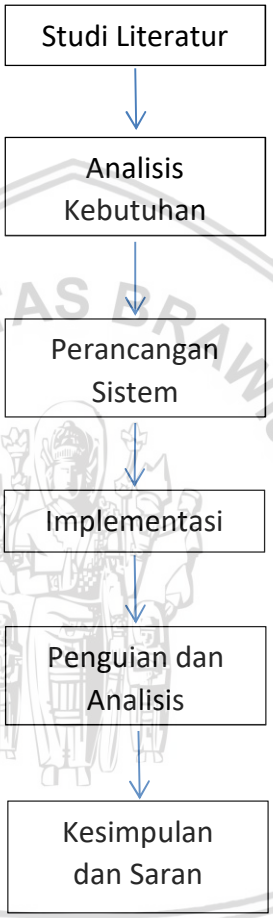
Gambar 2.20 LogoQ

Sumber: Denso (2013)

Jenis *QR Code* yang dapat menggabungkan fitur desain tingkat tinggi seperti ilustrasi, huruf dan logo. *QR Code* ini menggunakan Logika *Since proprietary* (Denso, 2013).

BAB 3 METODOLOGI

Bab ini akan menjelaskan alur dan metode yang akan dipakai, antara lain, literatur, fungsi algoritme AES, analisis kebutuhan, perancangan sistem, implementasi, pengujian dan analisis, dan pengambilan kesimulan dan saran. Gambar 3.1 menjelaskan mengenai diagram alir yang menjelaskan tentang runtutan pengerjaan penelitian:



Gambar 3.1 Digaram alir penelitian

Jenis penelitian yang digunakan pada penelitian ini adalah Implementatif-Pengembangan Lanjut (Enhancement). Menurut (Sugiyono, 2010) penelitian pengembangan merupakan metode penelitian yang digunakan untuk menghasilkan produk tertentu dan menguji keefektifan produk tersebut. Secara garis besar, penelitian pengembangan diawali dengan penelitianpenelitian skala kecil yang bisa dalam bentuk pengumpulan data terhadap permasalahan yang dihadapi dan ingin dicari solusinya.

3.1 Studi Literatur

Tahap ini merupakan tahap dimana dasar-dasar teori yang mendukung akan dikumpulkan sebagai sumber acuan untuk penelitian ini. Dasar-dasar teori yang akan digunakan, antara lain:

1. Proses enkripsi dan dekripsi
2. Algoritme AES 128 bit
3. PHP
4. QR code

3.2 Analisis Kebutuhan

Analisis kebutuhan perangkat lunak meliputi analisis spesifikasi perangkat lunak. Metode analisis menggunakan bahasa pemodelan UML.

Proses analisis kebutuhan dilakukan dengan menentukan kebutuhan-kebutuhan fungsional yang akan ada dalam aplikasi perangkat lunak yang akan dibuat, antara lain::

1. Sistem harus mampu menerima inputan semua jenis karakter berupa *string,* huruf, angka, dan simbol.
2. Sistem harus mampu merubah karakter menjadi QR *Code*.
3. Sistem harus mampu mengimplementasikan algoritme AES untuk enkripsi dan dekripsi pada QR- Code.

3.3 Perancangan Sistem

Perancangan arsitektur sistem yang digunakan dalam penelitian ini dilakukan setelah semua kebutuhan untuk pembuatan aplikasi yang didapatkan melalui tahap analisis kebutuhan telah terpenuhi. Perancangan dimulai dari perancangan umum sistem keamanan aplikasi dengan memodelkan gambaran terhadap lingkungan aplikasi, perancangan basis data dengan cara identifikasi *class*, perancangan alur aktifitas yang dimodelkan dalam *activity diagram* dan selanjutnya alur perancangan terhadap sisi keamanan aplikasi sebagai fokus penelitian.

3.4 Implementasi

Tahap implementasi diawali dengan penjabaran kondisi perangkat lunak pada saat akan diimplementasikan. Pembuatan aplikasi dikembangkan secara spesifik pada sistem operasi windows yang dibuat. Pada akhir tahap implementasi akan menghasilkan suatu aplikasi perangkat lunak yang siap untuk di uji dan dianalisis.

3.5 Pengujian dan Analisis

Pada bagain ini akan menjelaskan mengenai pengujian Perangkat lunak yang sesuai dengan kebutuhan-kebutuhan yang telah dirancang. Kemudian hasil pengujian ini akan dianalisis lebih lanjut. Terdapat 5 tahap pengujian dalam penelitian ini, antara lain:

- Pengujian *test vector*, dilakukan untuk mengetahui valid atau tidaknya program yang dibuat. Dari hasil output program akan dibandingkan dengan hasil manualisasi dari *test vector* algoritme AES.
- Pengujian Keamanan, dilakukan untuk memastikan privasi kode dalam QR *Code* terjaga. Dengan metode AES maka sistem keamanan aplikasi akan memberi perlindungan fisik pada pesan aslinya. Dengan metode ini juga akan meminimalkan celah bagi pihak lain untuk mengetahui isi informasi kode yang bukan termasuk kewenangannya.
- Pengujian waktu enkripsi dan dekripsi dilakukan untuk mengetahui waktu eksekusi sistem pada proses enkripsi dan dekripsi, dengan menggunakan fungsi microtime agar mengetahui waktu tiap *rounds* dan jumlah waktu keseluruhan menggunaka *key* dan *plaintext* sesuai *test vector*.
- Pengujian Fungsional, merupakan pengujian terhadap fungsional yang ada pada sistem dengan metode *black box* dengan tujuan untuk mengetahui apakah ada *error* atau ada fungsi yang tidak berjalan sesuai harapan.
- Pengujian *Non*-Fungsioanal untuk mengetahui karakteristik kualitas komponen atau sistem diuji. Non-fungsional mengacu pada aspek perangkat lunak yang mungkin tidak berhubungan dengan fungsi atau petunjuk tindakan tertentu seperti skalabilitas atau keamanan.

Pada akhir pengujian akan dilakukan analisis pembahasan terhadap hasil ujian yang telah dilakukan. Analisis ini diperlukan untuk mengetahui keberhasilan aspek valid, keamanan, waktu enkripsi dan dekripsi, fungsional dan *non*-fungsional dari enkripsi dan dekripsi yang memanfaatkan algoritme kriptografi AES.

3.6 Kesimpulan dan Saran

Pada bagian kesimpulan dan saran akan mencakup mengenai hasil analisis, implementasi, dan pengujian. Untuk bagian kesimpulan akan di jabarkan apakah sistem dapat menyelesaikan permasalahan-permasalahan yang telah di rumuskan. Untuk bagian saran berisi tentang saran-saran yang dapat menjadi pertimbangan untuk penelitian-penelitian berikutnya.

BAB 4 PERANCANGAN

Dalam bab ini terdapat 8 buah sub-bab utama yaitu algoritme AES, manualisasi enkripsi AES-128 bit, manualisasi dekripsi AES-128 bit, perancangan umum sistem, analisis kebutuhan, pemodelan kebutuhan, perancangan antarmuka, dan perancangan pengujian.

4.1 Algoritme AES

Dalam kriptografi, *Advanced Encryption Standard* (AES) merupakan standar enkripsi dengan tipe kunci-simetris. AES ini merupakan standar enkripsi yang digunakan oleh banyak negara di dunia. Didalam AES terdiri dari 3 *block cipher*, yaitu AES-128, AES-192 and AES-256, Masing-masing *cipher* memiliki ukuran 128-bit, dengan ukuran kuncinya sesuai dengan Namanya, misalnya AES-192 memiliki ukuran kunci 192 bit.

4.2 Manualisasi Enkripsi Aes 128 bit

Key: Thats my Kung Fu (16 karakter ASCII, 1 byte setiap karakter ASCII)

Konversi Dalam Hexa

Tabel 4.1 Konversi *Key* dalam bilangan hexadesimal

T	h	a	t	s		M	y		K	u	n	g		F	u
54	68	61	74	73	20	6D	79	20	4B	75	6E	67	20	46	75

Key dalam Hexa (128 bits): 54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75

Plaintext: **Two One Nine Two** (16 karakter ASCII, 1 byte setiap karakter ASCII)

Konversi Dalam Hexa :

Tabel 4.2 Konversi *plaintext* dalam bilangan hexadesimal

T	w	o		O	n	e		N	i	n	e		T	w	o
54	77	6F	20	4F	6E	65	20	4E	69	6E	65	20	54	77	6F

Plaintext dalam Hexa (128 bits): 54 77 6F 20 4F 6E 65 20 4E 69 6E 65 20 54 77 6F

a. Round key pertama

- Key Hexadecimal (128 bits):
  54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75
- *w*[0] = (54, 68, 61, 74),
- *w*[1] = (73, 20, 6*D*, 79),
- *w*[2] = (20, 4*B*, 75, 6*E*),
- *w*[3] = (67, 20,46, 75)
- *g*(*w*[3]):
  - Pergeseran kolom indek ke 3 *w*[3]: (20, 46, 75, 67)
  - Byte Subtitusi (S-Box): (*B*7, 5*A*, 9*D*, 85)

Menambahkan RCon (01, 00, 00, 00)

- $g(w[3])$ = (*B*6, 5*A*, 9*D*, 85)
- $w[4] = w[0] \oplus g(w[3])$ = (*E*2, 32, *FC*, *F* 1)
- $w[5] = w[4] \oplus w[1]$ = (91, 12, 91, 88)
- $w[6] = w[5] \oplus w[2]$ = (*B*1, 59, *E*4, *E*6)

$w[7] = w[6] \oplus w[3]$ = (*D*6, 79, *A*2, 93

Tabel 4.3 *First roundkey*

0101	0100	0110	1000	0110	0001	0111	0100
1011	0110	0101	1010	1001	1101	1000	0101
1110	0010	0011	0010	1111	1100	1111	0001
E2		32		FC		F1	

Hasil round key pertama : E2 32 FC F1 91 12 91 88 B1 59 E4 E6 D6 79 A2 93

b. ronde ke 0, *AddRoundKey*

- *state* matriks dan matriks *roundkey* ke 0

54 4F 4E 20	54 73 20 67
77 6E 69 54	68 20 4B 20
6F 6G 6E 77	61 6D 75 46
20 20 65 6F	74 79 6E 75

- xor kedua matriks 69⊕4B=22

0110 1001

0100 1011

0010 0010

- Matriks baru

00 3C 6E 47
1F 4E 22 74
0E 08 1B 31
54 59 0B 1A

c. Ronde ke 1, *Subtitution Bytes*

- *State* matriks

00 3C 6E 47
1F 4E 22 74
0E 08 1B 31
54 59 0B 1A

- Subtitusikan setiap masukan (byte) state matrix saat ini dengan mencocokkan inputan dalam AES S-Box. Misal 6E disubstitusikan dengan mencocokkan bari 6 dan E pada tabel S-Box, diperoleh 9F dan akan menghasilkan Matriks baru

63	EB	9F	A0
C0	2F	93	92
AB	30	AF	C7
20	CB	2B	A2

d. Ronde ke 1, *Shift Row*

- State Matriks

63	EB	9F	A0
C0	2F	93	92
AB	30	AF	C7
20	CB	2B	A2

- Menggeser baris 1 (1 kolom ke depan), baris 2 (2 kolom ke depan), baris 3 (3 kolom ke depan), kecuali baris 0. Berikut hasil dari pergeseran yang sudah dilakukan.

Baris ke 0	63	EB	9F	A0
Baris ke 1	2F	93	92	C0
Baris ke 2	AF	C7	AB	30
Baris ke 3	A2	20	CB	2B

e. Ronde ke 1, *Mix Column*

- Mix Column State Matrix

02	03	01	01		63	EB	9F	A0		BA	84	E8	1B
01	02	03	01	⊕	C0	2F	93	92	=	75	A4	8D	40
01	01	02	03		AB	30	AF	C7		F4	8D	06	7D
03	01	01	02		20	CB	2B	A2		7A	32	0E	5D

- Hasil *BA* merupakan hasil dari (02 • 63) ⊕ (03 • 2*F*) ⊕ (01 • *AF*) ⊕ (01 •*A*2):
  - (02 • 63) = 00000010 • 01100011 = 11000110
  - (03 • 2*F*) = (02 • 2*F*) ⊕ 2*F* = (00000010 • 00101111) ⊕ 00101111 = 01110001
  - (01 • *AF*) = *AF* = 10101111
  - (01 •*A*2) = *A*2 = 10100010

f. Ronde ke 1, *AddRoundKey*

- Matriks *State* dan Matriks *Roundkey* ke 1 :

BA 84 E8 1B	E2 91 B1 D6
75 A4 8D 40	32 12 59 79
F4 8D 06 7D	FC 91 E4 A2
7A 32 0E 5D	F1 88 E6 93

- Hasil XOR dari matriks *State* dan Matriks *Roundkey* ke 1

58 15 59 CD
47 B6 D4 39
08 1C E2 DF
8B BA E8 CE

g. Ronde ke 2

- Setelah *subtitusi byte* dan setelah pergeseran baris:

6A 59 CB BD	6A 59 CB BD
A0 4E 48 12	4E 48 12 A0
30 9C 98 9E	98 9E 30 9C
3D F4 9B 8B	8B 3D F4 9B

- Setelah *MixColumns* dan setelah *Roundkey*

15 C9 7F 9D	43 0E 09 3D
CE 4D 4B C2	C6 57 08 F8
89 71 BE 88	A9 C0 EB 7F
65 47 97 CD	62 C8 FE 37

h. ronde ke 3

- Setelah *subtitusi byte* dan setelah pergeseran baris:

1A AB 01 27	1A AB 01 27
B4 5B 30 41	5B 30 41 B4
D3 BA E9 D2	E9 D2 D3 BA
AA E8 BB 9A	9A AA E8 BB

- Setelah *MixColumns* dan setelah *Roundkey*

AA 65 FA 88	78 70 99 4B
16 0C 05 3A	76 76 3C 39
3D C1 DE 2A	30 7D 37 34
B3 4B 5A 0A	54 23 5B F1

i. ronde ke 4

- Setelah *subtitusi byte* dan setelah pergeseran baris:

BC 51 EE B3	BC 51 EE B3
38 38 EB 12	38 EB 12 38
04 FF 9A 18	9A 18 04 FF
20 26 39 A1	A1 20 26 39

- Setelah *MixColumns* dan setelah *Roundkey*

10 BC D3 F3	B1 08 04 E7
D8 94 E0 E0	CA FC B1 B2
53 EA 9E 25	51 54 C9 6C
24 40 73 7B	ED E1 D3 20

j. ronde ke 5

- Setelah *subtitusi byte* dan setelah pergeseran baris:

C8 30 F2 94	C8 30 F2 94
74 B0 C8 37	B0 C8 37 74
D1 20 DD 50	DD 50 D1 20
55 F8 66 B7	B7 55 F8 66

- Setelah *MixColumns* dan setelah *Roundkey*

2A 26 8F E9	9B 23 5D 2F
78 1E 0C 7A	51 5F 1C 38
1B A7 6F 0A	20 22 BD 91
5B 62 00 3F	68 F0 32 56

k. ronde ke 6

- Setelah *subtitusi byte* dan setelah pergeseran baris:

14 26 4C 15	14 26 4C 15
D1 CF 9C 07	CF 9C 07 D1
B7 93 7A 81	7A 81 B7 93
45 8C 23 B1	B1 45 8C 23

- Setelah *MixColumns* dan setelah *Roundkey*

A9 37 AA F2	14 8F C0 5E
AE D8 0C 21	93 A4 60 0F
E7 6C B1 9C	25 2B 24 92
F0 FD 67 3B	77 E8 40 75

l. ronde ke 7

- Setelah *subtitusi byte* dan setelah pergeseran baris:

FA 73 BA 58	FA 73 BA 58
DC 49 D0 76	49 D0 76 DC
3F F1 36 4F	36 4F 3F F1
F5 9B 09 9D	9D F5 9B 09

- Setelah *MixColumns* dan setelah *Roundkey*

9F 37 51 37	53 43 4F 85
AF EC 8C FA	39 06 0A 52
63 39 04 66	8E 93 3B 57
4B FB B1 D7	5D F8 95 BD

m. ronde ke 8

- Setelah *subtitusi byte* dan setelah pergeseran baris:

ED 1A 84 97	ED 1A 84 97
12 6F 67 00	6F 67 00 12
19 DC E2 5B	E2 5B 19 DC
4C 41 2A 7A	7A 4C 41 2A

- Setelah *MixColumns* dan setelah *Roundkey*

E8 8A 4B F5	66 70 AF A3
74 75 EE E6	25 CE D3 73
D3 1F 75 58	3C 5A 0F 13
55 8A 0C 38	74 A8 0A 54

n. ronde ke 9

- Setelah *subtitusi byte* dan setelah pergeseran baris:

33 51 79 0A	33 51 79 0A
3F 8B 66 8F	8B 66 8F 3F
EB BE 76 7D	76 7D EB BE
92 C2 67 20	20 92 C2 67

- Setelah *MixColumns* dan setelah *Roundkey*

B6 E7 51 8C	09 A2 F0 7B
84 88 98 CA	66 D1 FC 3B
34 60 66 FB	8B 9A E6 30
E8 D7 70 51	78 65 C4 89

o. ronde ke 10

- Setelah *subtitusi byte* dan setelah pergeseran baris:

01 3A 8C 21	01 3A 8C 21
33 3E B0 E2	3E B0 E2 33
3D B8 8E 04	8E 04 3D B8
BC 4D 1C A7	A7 BC 4D 1C

- Setelah *Roundkey* (perhatian : tidak ada *MixColumns* pada ronde terakhir)

29 57 40 1A
C3 14 22 02
50 20 99 D7
5F F6 B3 3A

4.3 Manualisasi Dekripsi Aes 128 bit

Key: Thats my Kung Fu (16 karakter ASCII, 1 byte setiap karakter ASCII)

Konversi Dalam Hexa

Tabel 4.4 Konversi Key dalam bilangan hexadesimal

T	h	a	t	s		m	y		K	u	n	g		F	u
54	68	61	74	73	20	6D	79	20	4B	75	6E	67	20	46	75

Key dalam Hexa (128 bits): 54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75

Plaintext: **Two One Nine Two** (16 karakter ASCII, 1 byte setiap karakter ASCII)

Konversi Dalam Hexa :

Tabel 4.5 Konversi Key dalam bilangan hexadesimal

T	w	o		O	n	e		N	i	n	e		T	w	o
54	77	6F	20	4F	6E	65	20	4E	69	6E	65	20	54	77	6F

Plaintext dalam Hexa (128 bits): 54 77 6F 20 4F 6E 65 20 4E 69 6E 65 20 54 77 6F

a. *Roundkey* pertama

- Key pada Hex (128 bits): 54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75
- $w[0] = (54, 68, 61, 74), w[1] = (73, 20, 6D, 79), w[2] = (20, 4B, 75, 6E), w[3] = (67, 20, 46, 75)$
- $g(w[3])$:
  - pergeseran *circular* ke kiri dari w[3]: (20, 46, 75, 67)
  - *Subtitusi byte* (S-Box): (B7, 5A, 9D, 85)
  - Menambahkan putaran konstan (01, 00, 00, 00) diketahui: $g(w[3])$ = (B6, 5A, 9D, 85)

p. $w[4] = w[0] \oplus g(w[3])$ = (E2, 32, FC, F1):

- $w[5] = w[4] \oplus w[1] = (91, 12, 91, 88), w[6] = w[5] \oplus w[2] = (B1, 59, E4, E6)$, $w[7] = w[6] \oplus w[3]$ = (D6, 79, A2, 93)

0101 0100 1011 0110 1110 0010	0110 1000 0101 1010 0011 0010	0110 0001 1001 1101 1111 1100	0111 0100 1000 0101 1111 0001
E2	32	FC	F1

- Roundkey pertama: E2 32 FC F1 91 12 91 88 B1 59 E4 E6 D6 79 A2 93

b. Hasil ekspansi key 10 putaran

- Round 0: 54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75
- Round 1: E2 32 FC F1 91 12 91 88 B1 59 E4 E6 D6 79 A2 93
- Round 2: 56 08 20 07 C7 1A B1 8F 76 43 55 69 A0 3A F7 FA
- Round 3: D2 60 0D E7 15 7A BC 68 63 39 E9 01 C3 03 1E FB
- Round 4: A1 12 02 C9 B4 68 BE A1 D7 51 57 A0 14 52 49 5B
- Round 5: B1 29 3B 33 05 41 85 92 D2 10 D2 32 C6 42 9B 69
- Round 6: BD 3D C2 B7 B8 7C 47 15 6A 6C 95 27 AC 2E 0E 4E
- Round 7: CC 96 ED 16 74 EA AA 03 1E 86 3F 24 B2 A8 31 6A
- Round 8: 8E 51 EF 21 FA BB 45 22 E4 3D 7A 06 56 95 4B 6C
- Round 9: BF E2 BF 90 45 59 FA B2 A1 64 80 B4 F7 F1 CB D8
- Round 10: 28 FD DE F8 6D A4 24 4A CC C0 A4 FE 3B 31 6F 26

c. *Add RoundKey,* Round ke 10 (Menggunakan roundkey 10 dari hasil ekspansi *key*)

q. Matriks *State* dan *Roundkey* 10

29 57 40 1A C3 14 22 02 50 20 99 D7 5F F6 B3 3A	28 6D CC 3B FD A4 C0 31 DE 24 A4 6F F8 4A FE 26

- Hasil matriks

01 3A 8C 21
3E B0 E2 33
8E 04 3D B8
A7 BC 4D 1C

d. Round ke 9, *Invers* dari subtitusi bytes

- Subtitusikan matriks dengan data pada tabel *invers* S-BOX

01 3A 8C 21
3E B0 E2 33
8E 04 3D B8
A7 BC 4D 1C

- Hasil matriks baru

09 A2 F0 7B
D1 FC 3B 66
E6 30 8B 9A
89 78 65 C4

e. Round ke 9, *Invers* dari pergeseran baris

- Matriks (Hasil dari proses subtitusi)

09 A2 F0 7B
D1 FC 3B 66
E6 30 8B 9A
89 78 65 C4

- Hasil matriks (Proses pergeseran sama halnya dengan enkripsi)

09 A2 F0 7B
66 D1 FC 3B
8B 9A E6 30
78 65 C4 89

f. *Add RoundKey,* Round ke 9 (Menggunakan roundkey 1 dari hasil ekspansi key)

- Matriks *State* dan *Roundkey* (Melakukan *XOR* dengan *key* ke 9)

09 A2 F0 7B
66 D1 FC 3B
8B 9A E6 30
78 65 C4 89

$\oplus$

BF 45 A1 F7
E2 59 64 F1
BF FA 80 CB
90 B2 B2 D8

- Hasil matriks setelah proses XOR

B6 E7 51 8C
84 88 98 CA
34 60 66 FB
E8 D7 70 51

g. Round ke 9, *Invers* dari *Mix Column*

- Setelah *invers* dari *Mix Column*

$$\begin{pmatrix} B6 & E7 & 51 & 8C \\ 84 & 88 & 98 & CA \\ 34 & 60 & 66 & FB \\ E8 & D7 & 70 & 51 \end{pmatrix} \times \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 03 & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} = \begin{pmatrix} 33 & 51 & 79 & 0A \\ 8B & 66 & 8F & 3F \\ 76 & 7D & EB & BE \\ 20 & 92 & C2 & 67 \end{pmatrix}$$

h. Round ke 8

- Setelah *invers subtitusi byte* dan pergeseran baris

66 70 AF A3	66 70 AF A3
CE D3 73 25	25 CE D3 73
0F 13 3C 5A	3C 5A 0F 13
54 74 A8 0A	74 A8 0A 54

- Hasil setelah *Roundkey* dan setelah *Mix Column*

E8 8A 4B F5	ED 1A 84 97
74 75 EE E6	6F 67 00 12
D3 1F 75 58	E2 5B 19 DC
55 8A 0C 38	7A 4C 41 2A

i. Round ke 7

- Setelah *invers subtitusi byte* dan pergeseran baris

53 43 4F 85	53 43 4F 85
06 0A 52 39	39 06 0A 52
3B 57 8E 93	8E 93 3B 57
BD 5D F8 95	5D F8 95 BD

- Hasil setelah *Roundkey* dan setelah *Mix Column*

9F 37 51 37	0A 73 BA 58
AF EC 8C FA	49 D0 76 DC
63 39 04 66	36 4F 3F F1
4B FB B1 D7	9D F5 9B 09

j. Round ke 6

- Setelah *invers subtitusi byte* dan pergeseran baris

14 8F C0 5E	14 8F C0 5E
A4 60 0F 93	93 A4 60 0F
24 92 25 2B	25 2B 24 92
75 77 E8 40	77 E8 40 75

- Hasil setelah *Roundkey* dan setelah *Mix Column*

A9 37 AA F2	14 26 4C 15
AE D8 0C 21	CF 9C 07 D1
E7 6C B1 9C	7A 81 B7 93
F0 FD 67 3B	B1 45 8C 23

k. Round ke 5

- Setelah *invers subtitusi byte* dan pergeseran baris

9B 23 5D 2F	9B 23 5D 2F
5F 1C 38 51	51 5F 1C 38
BD 91 20 22	20 22 BD 91
56 68 F0 32	68 F0 32 56

- Hasil setelah *Roundkey* dan setelah *Mix Column*

2A 26 8F E9	C8 30 F2 94
78 1E 0C 7A	B0 C8 37 74
1B A7 6F 0A	DD 50 D1 20
5B 62 00 3F	B7 55 F8 66

l. Round ke 4

- Setelah *invers subtitusi byte* dan pergeseran baris

B1 08 04 E7	B1 08 04 E7
FC B1 B2 CA	CA FC B1 B2
C9 6C 51 54	51 54 C9 6C
20 ED E1 D3	ED E1 D3 20

- Hasil setelah *Roundkey* dan setelah *Mix Column*

10 BC D3 F3	BC 51 EE B3
D8 94 E0 E0	38 EB 12 38
53 EA 9E 25	9A 18 04 FF
24 40 73 7B	A1 20 26 39

m. Round ke 3

- Setelah *invers subtitusi byte* dan pergeseran baris

78 70 99 4B	78 70 99 4B
76 3C 39 76	76 76 3C 39
37 34 30 7D	30 7D 37 34
F1 54 23 5B	54 23 5B F1

- Hasil setelah *Roundkey* dan setelah *Mix Column*

AA 65 FA 88	1A AB 01 27
16 0C 05 3A	5B 30 41 B4
3D C1 DE 2A	E9 D2 D3 BA
B3 4B 5A 0A	9A AA E8 BB

n. Round ke 2

- Setelah *invers subtitusi byte* dan pergeseran baris

43 0E 09 3D	43 0E 09 3D
57 08 F8 C6	C6 57 08 F8
EB 7F A9 C0	A9 C0 EB 7F
37 62 C8 FE	62 C8 FE 37

- Hasil setelah *Roundkey* dan setelah *Mix Column*

15 C9 7F 9D	6A 59 CB BD
CE 4D 4B C2	4E 48 12 A0
89 71 BE 88	98 9E 30 9C
65 47 97 CD	8B 3D F4 9B

o. Round ke 1

- Setelah *invers subtitusi byte* dan pergeseran baris

58 15 59 CD	58 15 59 CD
B6 D4 39 47	47 B6 D4 39
E2 DF 08 1C	08 1C E2 DF
CE 8B BA E8	8B BA E8 CE

- Hasil setelah *Roundkey* dan setelah *Mix Column*

BA 84 E8 1B	63 EB 9F A0
75 A4 8D 40	2F 93 92 C0
F4 8D 06 7D	AF C7 AB 30
7A 32 0E 5D	A2 20 CB 2B

p. Round Final

- Setelah *invers subtitusi byte* dan pergeseran baris

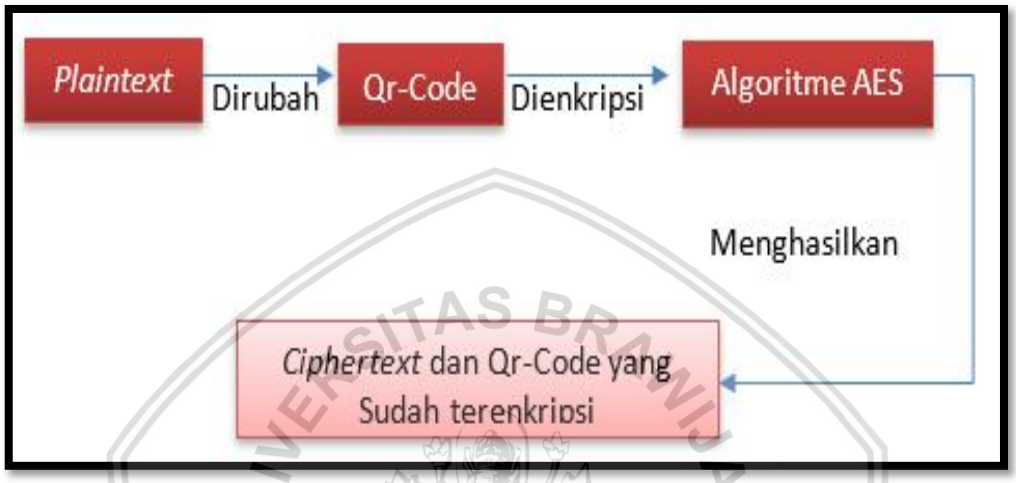
FB E9 DB E0	00 3C 6E 47
15 DC 4F BA	1F 4E 22 74
79 C6 62 04	0E 08 1B 31
3A B7 1F F1	54 59 0B 1A

- Hasil setelah *Roundkey* (tidak ada proses *MixColumn*)

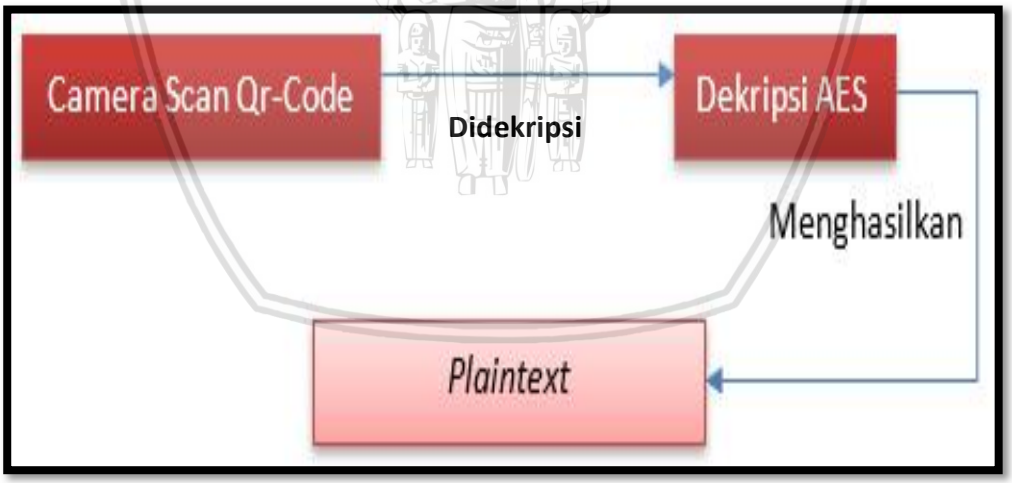
54 4F 4E 20
77 6E 69 54
6F 65 6E 77
20 20 65 6F

4.4 Perancangan Umum Sistem

Perancangan umum sistem adalah tahap awal dalam perancangan sistem/aplikasi perangkat lunak. Perancangan ini bertujan untuk merepresentasikan arsitektur aplikasi/sistem secara umum, seperti yang ditunjukkan pada Gambar 4.1 dan Gambar 4.2.



Gambar 4.1 Perancangan Enkripsi



Gambar 4.2 Perancangan Dekripsi

Pada Gambar 4.1 merupakan rancangan enkripsi yang berfungsi untuk merubah dari kode asli menjadi *QR Code* dan dienkripsi AES setelah itu menghasilkan ciphertext dan *QR Code* yang telah dienkripsi. Selanjutnya pada Gambar 4.2 berfungsi sebagai dekripsi yang bermula dari gambar *QR Code* di scan lalu dimasukkan ke dalam dekripsi AES, dekripsi ini berfungsi untuk membalikkan Kode asli yang terenkripsi menjadi kode asli atau plaintext.

4.5 Analisis Kebutuhan

Bagian analisis kebutuhan menjelaskan kebutuhan fungsional, non-fungsional, spesifikasi dan manajemen kebutuhan. Analisis kebutuhan Fungsional merupakan bagian yang penting dalam perancangan sistem karena menggambarkan aplikasi yang akan dibuat dan berguna sebagai dokumentasi yang akan berguna pengembangan aplikasi selanjutnya.

4.5.1 Kebutuhan Fungsional

1. Sistem harus mampu menerima inputan berupa kode atau key.
2. Sistem harus mampu menerima inputan berupa kode atau *plaintext*.
3. Sistem harus mampu melakukan enkripsi dengan algoritmea aes.
4. Sistem harus mampu mengubah *plaintext* menjadi *QR Code*.
5. Sistem harus mampu melakukan Scan *QR Code*.
6. Sistem harus mampu melakukan dekripsi dengan algoritmea aes.

4.5.2 Kebutuhan Non-Fungsional

1. Sistem harus dapat berjalan pada semua web browser modern (Portability).

4.5.3 Spesifikasi dan Manajemen Kebutuhan

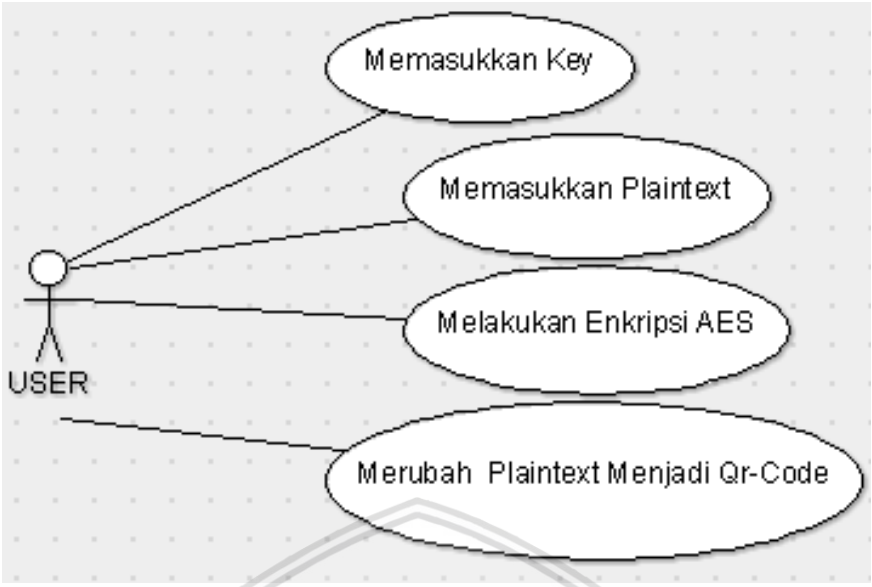
1. Memasukkan *key* (AES_1_001)
2. Memasukan *plaintext* (AES_1_002)
3. Melakukan Enkripsi AES (AES_1_003)
4. Merubah *plaintext* Menjadi *QR Code* (AES_1_004)
5. Melakukan *Scan QR Code* (AES_1_005)
6. Melakukan Dekripsi AES (AES_1_006)

4.6 Pemodelan Kebutuhan

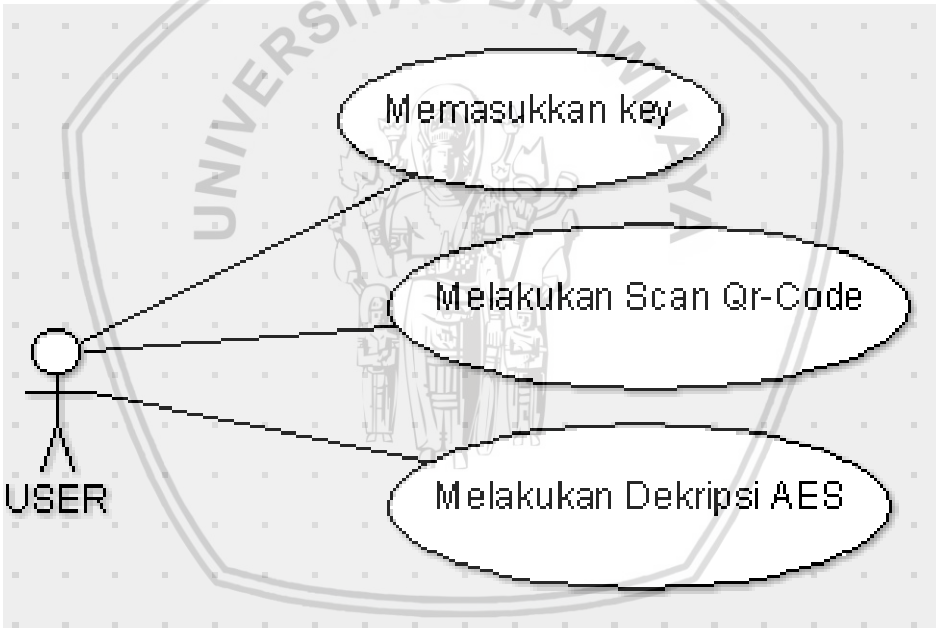
Pada sub-bab Pemodelan kebutuhan akan menjelaskan 5 buah poin yaitu: *use case* diagram, *use case* scenario, *activity* diagram, *sequence* diagram dan *class* diagram.

4.6.1 *Use Case* Diagram

Use case Diagram merupakan salah satu model UML yang digunakan untuk mendeskripsikan kebutuhan fungsional sebuah sistem dari perspektif *end user*. *Use case* juga menunjukan perilaku (*behavior*) dari sistem yang akan dibuat. Diagram *use case* pada sistem ini dapat dilihat dalam Gambar 4.3 dan 4.4.



Gambar 4.3 *Use case* diagram enkripsi



Gambar 4.4 *Use case* diagram dekripsi

Diagram *use case* pada Gambar 4.3 meupakan enkripsi yang mana user dapat memasukkan *key,* memasukkan *plaintext,* melakukan enkripsi AES dan merubah plaintext menjadi *QR Code* dan diagram *use case* pada Gambar4.4 meupakan dekripsi yang mana user dapat memasukkan *key,* melakukan scan *QR Code*, dan melakukan dekripsi AES.

4.6.2 Use Case Skenario

Skenario *use case* berguna untuk menunjukan kemungkinan yang terjadi dalam interaksi antara aktor/pelaku dengan sistem. Hal ini meliputi bagaimana keadaan yang terjadi sebelum pelaku/aktor memberikan perintah, apa saja

perintah yang aktor/pelaku berikan, dan bagaimana respon sistem terhadap perintah yang diberikan. Berikut adalah *use case* Skenario yang terdapat pada aplikasi yang akan diujikan.

Use case Skenario Memasukkan *Key* (AES_1_001)

Tabel 4.6 *Use case* Skenario memasukkan *key*

Actor	User
Objective	User dapat memasukkan inputan berupa *key*..
Pre-condition	User telah mengakses website AES.
Main flow	1. User memilih menu "Enkripsi" atau "Dekripsi". 2. Sistem menampilkan halaman Enkripsi atau Dekripsi. 3. User mengimputkan *key* yang panjang karakter harus 16 karakter.
Alternative flows	3.a) Jika user mengimputkan key melebihi 16 karakter sistem akan menampilkan pesan "BATASAN KEY MAKSIMAL 16 KARAKTER"
Post-condition	*Key* telah diinputkan ke dalam sistem.

Pada tabel 4.6 merupakan *use case* skenario atau langkah-langkah memasukkan *key* ke dalam sistem AES.

Use case Skenario Memasukkan *Plaintext* (AES_1_002)

Tabel 4.7 *Use case* skenario memasukkan *plaintext*

Actor	User
Objective	User dapat memasukkan inputan berupa *plaintext.*
Pre-condition	User telah mengakses website AES.
Main flow	1. User memilih menu "Enkripsi". 2. Sistem menampilkan halaman Enkripsi. 3. User mengimputkan Plaintext atau Kode yang di inginkan.
Alternative flows	-
Post-condition	*Plaintext* telah diinputkan ke dalam sistem.

Pada tabel 4.7 merupakan *use case* skenario atau langkah-langkah memasukkan *plaintext* ke dalam sistem AES.

Use case Skenario Melakukan Enkripsi (AES_1_003)

Tabel 4.8 *Use case* skenario melakukan enkripsi

Actor	User
Objective	User dapat melakukan Enkripsi
Pre-condition	User telah mengakses website AES.
Main flow	1. User memilih menu “Enkripsi”. 2. Sistem menampilkan halaman Enkripsi. 3. User mengimputkan *key dan plaintext.* 4. Sistem merubah *plaintext* menjadi *QR Code*. 5. User mengklik tombol “Encrypt”. 6. Sistem menampilkan hasil proses enkripsi mulai dari round 1-10 dan *QR Code*.
Alternative flows	-
Post-condition	*QR Code* telah dienkripsi oleh sistem.

Pada tabel 4.8 merupakan *use case* skenario atau langkah-langkah melakukan enkripsi *QR Code*.

Use case Skenario merubah *plaintext* menjadi *QR Code* (AES_1_004)

Tabel 4.9 *Use case* skenario merubah *plaintext* menjadi *QR Code*

Actor	User
Objective	User dapat merubah *plaintext* menjadi *QR Code*.
Pre-condition	User telah mengimputkan *key dan Plaintext.*
Main flow	1. User mengklik tombol “Lihat Kode QR Plaintext”. 2. Sistem akan menampilkan gambar *QR Code*
Alternative flows	-
Post-condition	Sistem telah menampilkan gambar *QR Code* kepada user.

Pada tabel 4.9 merupakan *use case* skenario atau langkah-langkah merubah *plaintext* menjadi *QR Code*.

Use case Skenario melakukan scan *QR Code* (AES_1_005)

Tabel 4.10 *Use case* skenario melakukan scan *QR Code*

Actor	User
Objective	User dapat melakukan Scan *QR Code*.
Pre-condition	User telah melakukan proses enkripsi kode yang di ingnkan.

Main flow	1. User memilih menu dekripsi. 2. Sistem menampilkan halaman dekripsi. 3. User mengklik tombol “Nyalakan” dan memassukkan *key.* 4. Sistem akan mengaktifkan fungsi kamera pada perangkat yang digunakan. 5. User menunjukan gambar dari *QR Code* yang telah dienkripsi sebelumnya.
Alternative flows	-
Post-condition	Gambar *QR Code* telah terbaca oleh sistem.

Pada tabel 4.10 merupakan *use case* skenario atau langkah-langkah melakukan scan *QR Code*.

Use case Skenario melakukan Dekripsi (AES_1_006)

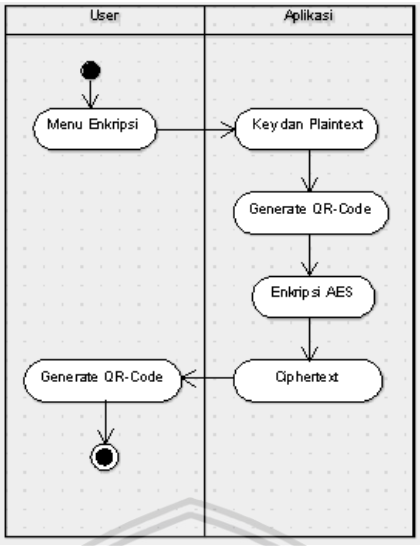
Tabel 4.11 *Use case* skenario melakukan dekripsi

Actor	User
Objective	User dapat melakukan Dekripsi.
Pre-condition	User telah mengakses halaman Dekripsi.
Main flow	1. User memilih menu dekripsi. 2. Sistem menampilkan halaman dekripsi. 3. User mengklik tombol “Nyalakan” dan memassukkan *key.* 4. Sistem akan mengaktifkan fungsi kamera pada perangkat yang digunakan. 5. User menunjukan gambar dari *QR Code* yang telah dienkripsi sebelumnya. 6. Sistemakan menampilkan proses dari dekripsi mulai dari state awal sampai akhir.
Alternative flows	-
Post-condition	Plain atau kode telah didekripsi oleh sistem.

Pada tabel 4.11 merupakan *use case* skenario atau langkah-langkah melakukan dekripsi *QR Code*.

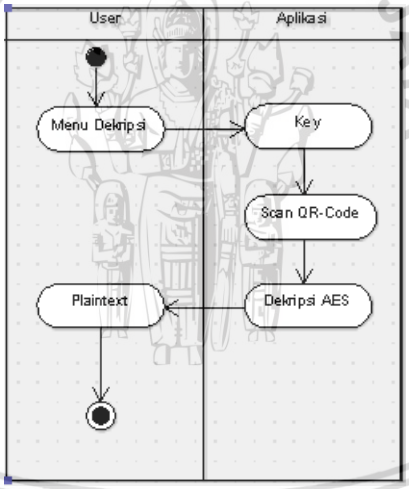
4.6.3 *Activity* Diagram

Activity diagram adalah suatu diagram yang menggambarkan alur dari aktivitas-aktivitas yang ada dalam suatu sistem. Perbedaan antara *activity diagram* dan *use case* diagram adalah, *use case* diagram menggambarkan bagaimana pelaku atau actor berinteraksi dengan sistem. Sedangkan *activity diagram* menggambarkan jalannya proses interaksi di dalam sistem.



Gambar 4.5 *Activity* diagram enkripsi

Pada Gambar 4.5 *user* harus menekan tombol enkripsi lalu aplikasi meminta *key* dan *plaintext* untuk di rubah menjadi *QR Code* kemudian dienkripsi sehingga menghasilkan *ciphertext* yang akan di *generate* kedalam *QR Code*.



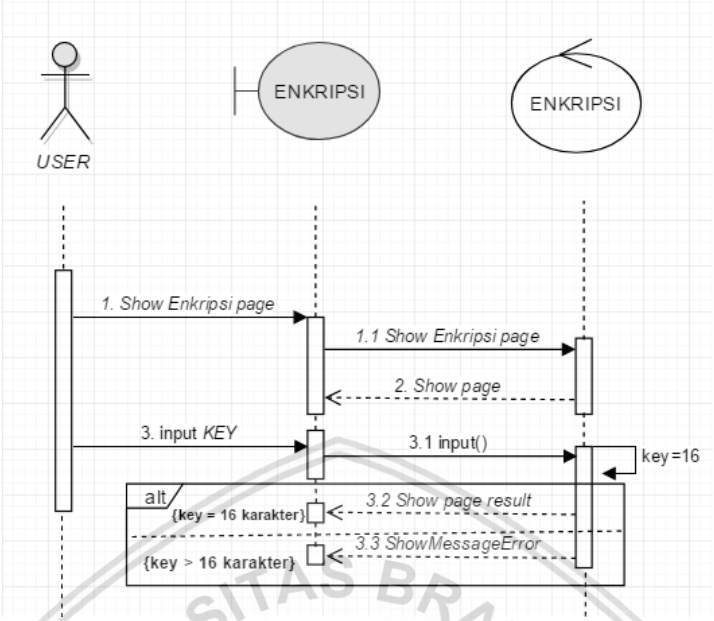
Gambar 4.6 *Activity* diagram dekripsi

Pada Gambar 4.6 *user* harus menekan tombol dekripsi lalu memasukkan *key dan* aplikasi meminta scan *QR Code* untuk di dekripsi dan sistem menampilkan *plaintext.*

4.6.4 *Sequence* Diagram

Diagram *sequence* (urutan) adalah diagram yang menjelaskan pertukaran pesan dalam rentang waktu tertentu. Diagram s*equence* ini berguna untuk merepresentasikan langkah-langkah yang akan di jalankan oleh sistem sebagai respon dari suatu aksi/kejadian yang dapat menghasilkan suatu *output* tertentu.

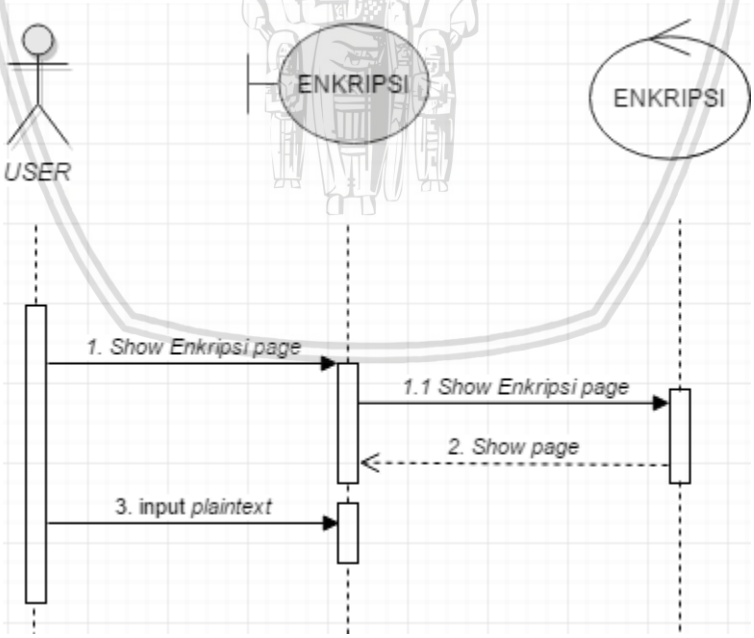
Sequence diagram memasukkan *key* (AES_1_001)



Gambar 4.7 *Sequence* diagram memasukkan *key*

pada gambar 4.7 merupakan *sequence* diagram memasukkan *key* atau langkah-langkah memasukkan *key* yang dilakukan pada sistem.

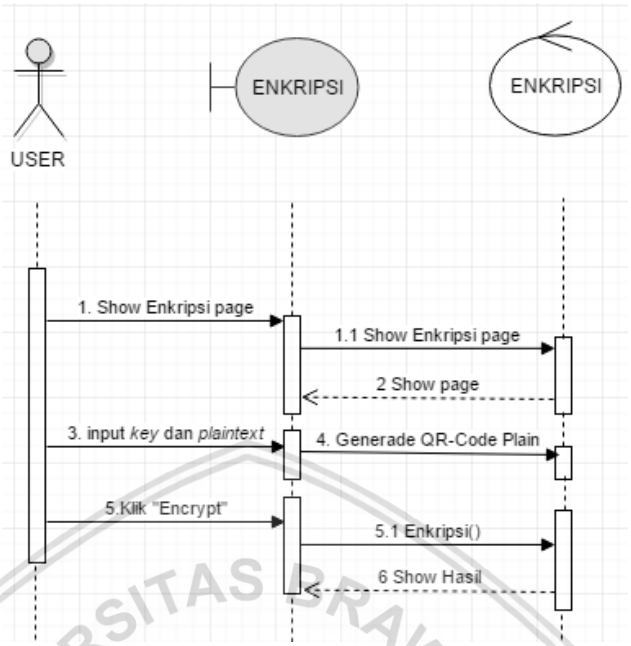
Sequence diagram memasukkan *plaintext* (AES_1_002)



Gambar 4.8 *Sequence* diagram memasukkan *plaintext*

pada gambar 4.8 merupakan s*equence* diagram memasukkan *plaintext* atau langkah-langkah memasukkan *plaintext* yang dilakukan pada sistem.

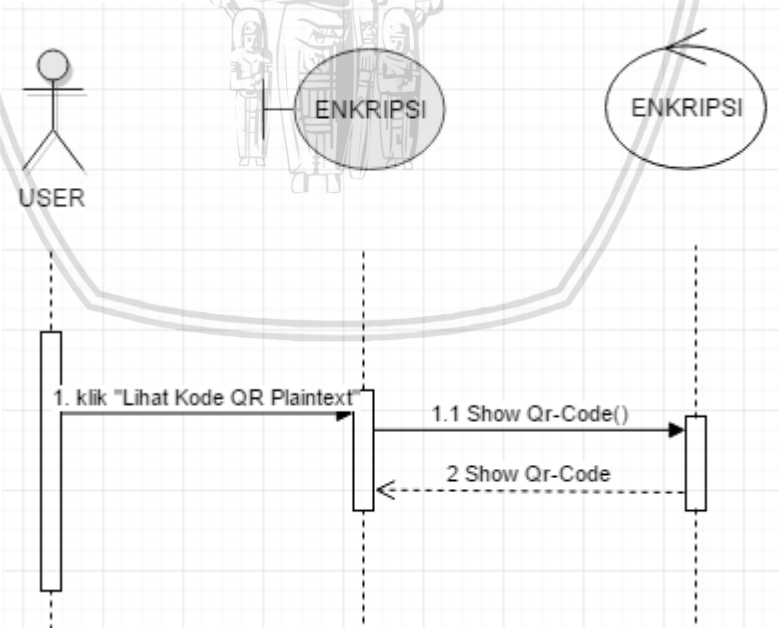
Sequence diagram melakukan enkripsi (AES_1_003)



Gambar 4.9 *Sequence* diagram melakukan enkripsi

pada gambar 4.9 merupakan s*equence* diagram melakukan enkripsi *QR Code* atau langkah-langkah enkripsi *QR Code* yang dilakukan pada sistem.

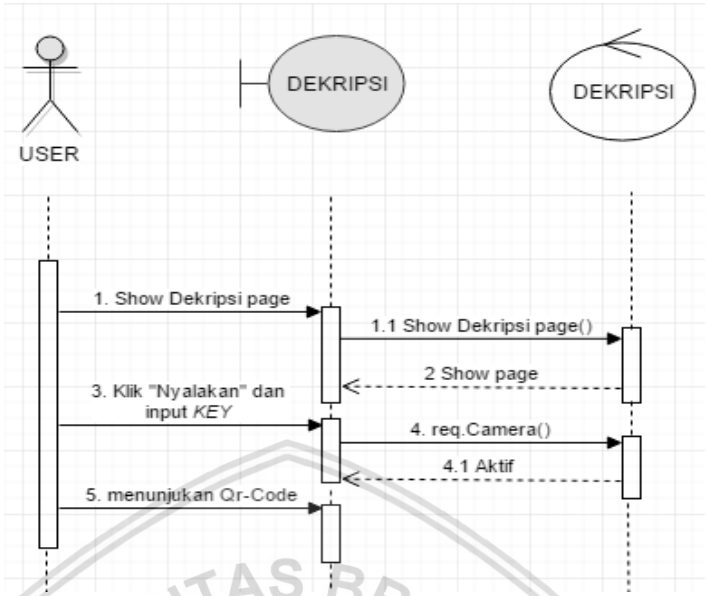
Sequence diagram merubah *plaintext* menjadi *QR Code* (AES_1_004)



Gambar 4.10 *Sequence* diagram merubah *plaintext* menjadi QR Code

pada gambar 4.10 merupakan s*equence* diagram merubah *plaintext* menjadi *QR Code* atau langkah-langkah merubah *plaintext* menjadi *QR Code* yang dilakukan pada sistem.

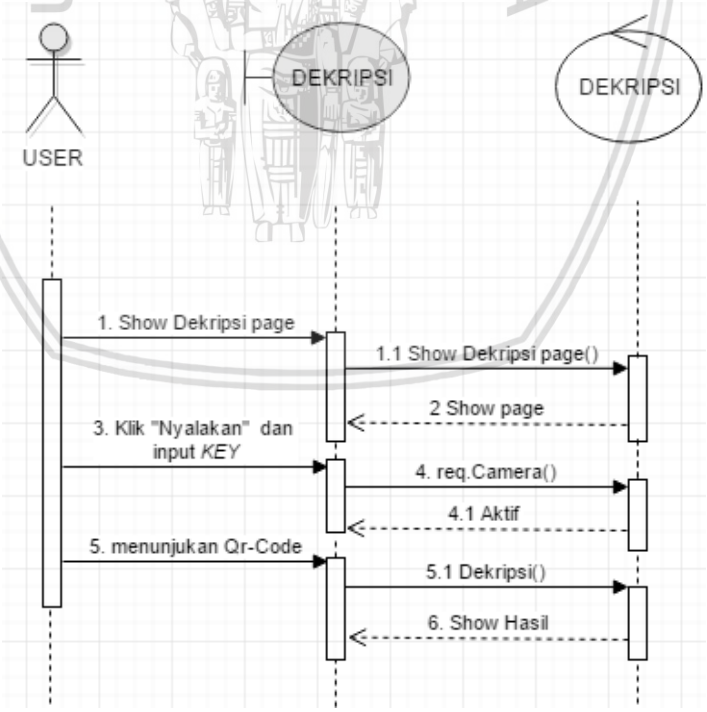
Sequence diagram melakukan scan *QR Code* (AES_1_005)



Gambar 4.11 *Sequence* diagram melakukan scan *QR Code*

pada gambar 4.11 merupakan *sequence* diagram melakukan scan *QR Code* atau langkah-langkah melakukan scan *QR Code* yang dilakukan pada sistem.

Sequence diagram melakukan dekripsi (AES_1_006)

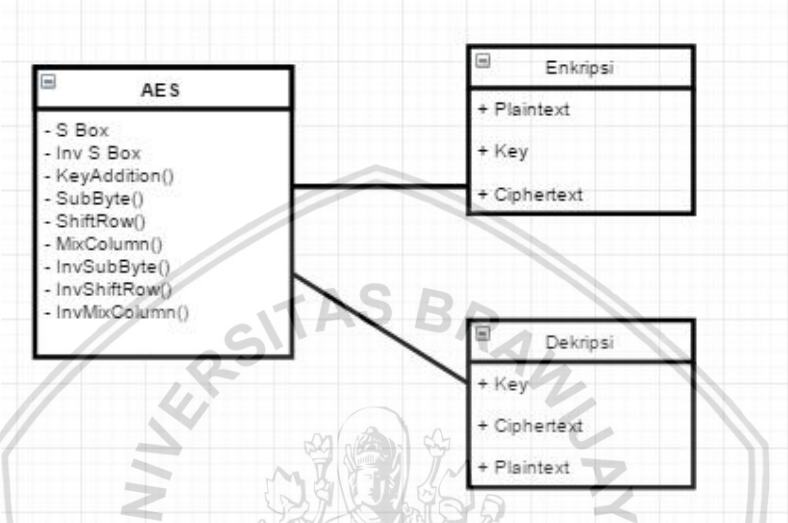


Gambar 4.12 *Sequence* diagram melakukan dekripsi

pada gambar 4.12 merupakan *sequence* diagram melakukan dekripsi QR *Code* atau langkah-langkah melakukan dekripsi QR *Code* yang dilakukan pada sistem.

4.6.5 *Class* Diagram

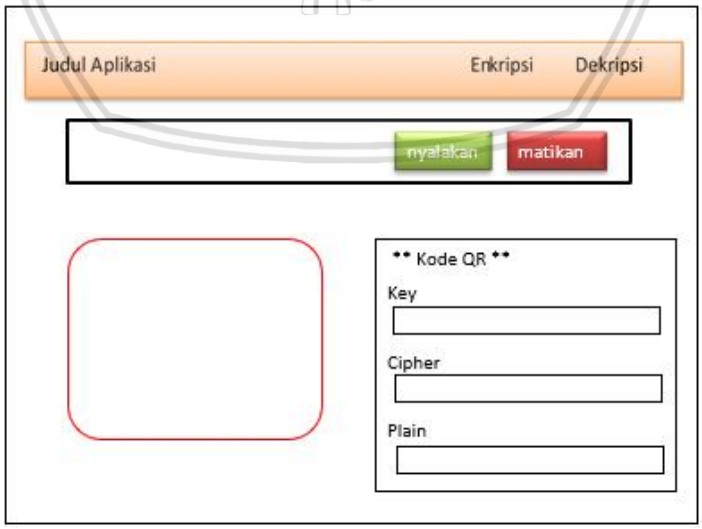
Pada gambar 4.13 merupakan gambar *class* diagram dari aplikasi AES yang menunjukkan relasi serta himpunan antar kelas yang ada. Kelas-kelas yang dimaksud adalah AES, enkripsi, dan dekripsi. Kelas AES berisi *method-method* sebagai berikut : S Box, Inv S Box, *KeyAddition(), SubByte(), ShiftRow(), MixColumn(),* dll. Lalu pada kelas enkripsi dan dekripsi berisi *method, plaintext, key, ciphertext*



Gambar 4.13 *Class* diagram

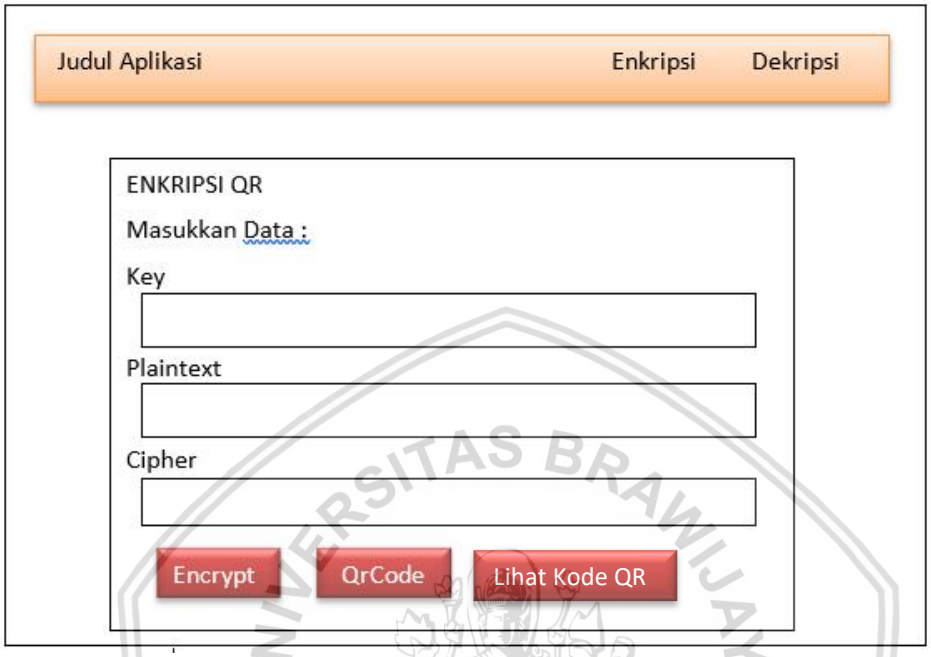
4.7 Perancangan Antarmuka

Perancangan antarmuka merupakan sebuah rancangan untuk membuat suatu media prantara interaksi antara sistem dan pengguna.



Gambar 4.14 Perancangan antarmuka dekripsi

Pada gambar 4.14 merupaka rancangan *interface* dari menu dekripsi dimana user menginputkan *key* lalu mengklik tombol nyalakan untuk mengaktifkan kamera pada perangkat yang nantinya kamera tersebut digunakan untuk proses scan *QR Code* dan sistem akan langsung memproses hasil dari proses scan tadi.



Gambar 4.15 Perancangan antarmuka enkripsi

Pada gambar 4.15 merupakan rancangan *interface* dari menu enkripsi dimana user mengimputkan *key*, *plaintext* lalu mengklik tombol *encrypt* untuk memulai proses enkripsi dan menklik tombol *QR Code* untuk menampilkan gambar *QR Code* atau lihat kode QR *plaintext* untuk melihat *QR Code* dari *plaintext*.

4.8 Perancangan Pengujian

Pengujian pada penelitian ini menggunakan 5 buah metode yaitu, pengujian *test vectors,* pengujian keamanan, pengujian waktu enkripsi dan dekripsi, dan pengujian *fungsional* dan *non-fungsional.* Pengujian *test vectors* dimulai dengan memasukkan sebuah *Key* dan sebuah *Plaintext* yang didapat dari jurnal menurut Adam Barent (2014). Hasil yang didapat dari *output* program kemudian akan di bandingkan dengan hasil dari proses manualisasi *test vectors*. Pada pengujian keamanan dilakukan dengan menggunakan aplikasi yang bernama "Quick Scan" untuk menscan *QR Code* yang sudah di enkripsi maupun belum dienkripsi. Pada pengujian waktu enkripsi dan dekripsi dilakukan dengan memasukkan sebuah *plaintext* dan *key* sesuai dengan *test vectors* AES. Untuk pengujian *fungsional* dan *non-fungsional* dilakukan dengan metode *black box testing* dengan mencoba seluruh fungsional yang ada pada sistem.

BAB 5 IMPLEMENTASI DAN PENGUJIAN

5.1 Implementasi Algoritme AES

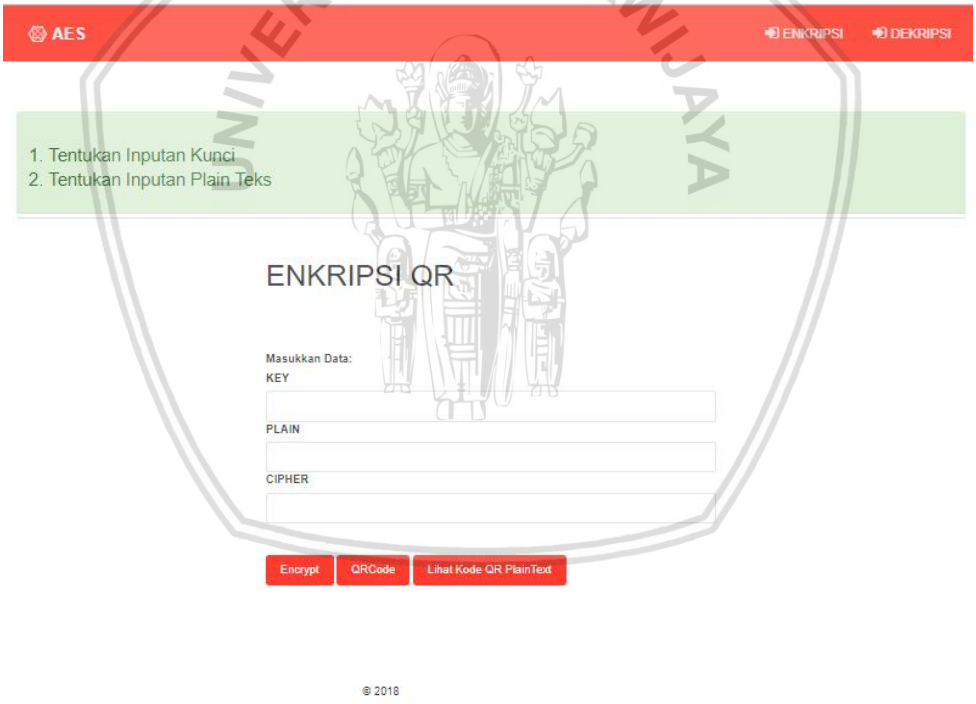
Pada sub-bab ini akan membahas tentang Implementasi algoritme AES pada proses enkripsi dan dekripsi *QR Code*. Untuk kode sumber (*source code)* dapat dilihat pada lampiran A.

5.2 Implementasi Antarmuka

Antarmuka atau *interface* adalah suatu media prantara interaksi antara sistem dan pengguna. Pada proses implementasi *interface* untuk penelitian ini dibagi menjadi dua yaitu, *interface dekripsi dan interface enkripsi*.

5.2.1 Implementasi Halaman Enkripsi

Halaman enkripsi merupakan *intrface* bagi pengguna untuk melakukan proses enkripsi pada *plaintext* menjadi *QR Code* dalam sistem AES Enkripsi Dekripsi.

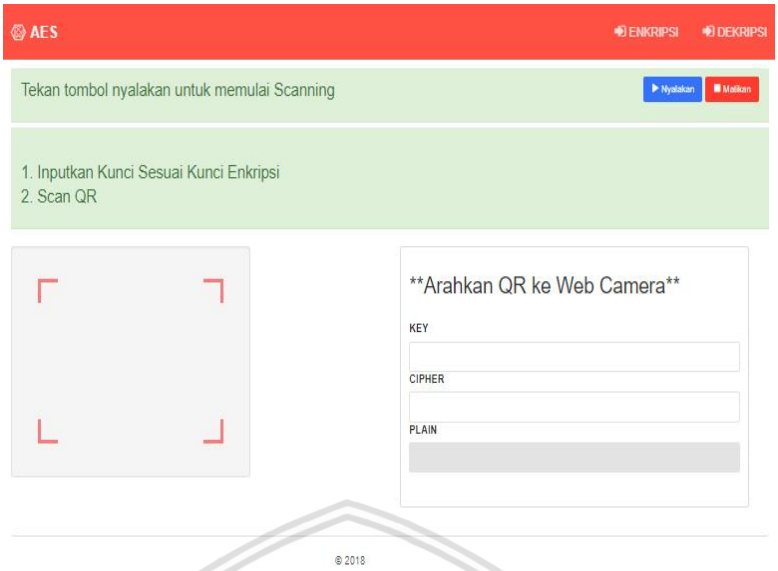


Gambar 5.1 Tampilan halaman enkripsi

Pada gambar 5.1 merupakan gambar dari implementasi tampilan halaman Enkripsi.

5.2.2 Implementasi Halaman Dekripsi

Halaman dekripsi merupakan *intrface* bagi pengguna untuk melakukan proses dekripsi pada *QR Code* yang ingin dioalah menjadi *plaintext* mengunakan sistem AES Enkripsi Dekripsi.



Gambar 5.2 Tampilan halaman dekripsi

Pada gambar 5.2 merupakan gambar dari implementasi tampilan halaman Dekripsi.

5.3 Pengujian

Proses pengujian terdiri dari 5 yaitu pengujian *test vectors,* pengujian keamanan, pengujian waktu enkripsi dan dekripsi, pengujian kebutuhan fungsional dan *non*-fungsional. Pada pengujian *test vectors* dilakukan dengan memasukkan sebuah *plaintext* dan sebuah *key* yang didapatkan dari jurnal menurut Adam Barent (2014). Hasil yang didapat dari *output* program kemudian akan di bandingkan dengan hasil dari proses manualisasi *test vectors.* Pada pengujian keamanan dilakukan dengan menggunakan aplikasi yang bernama "Quick Scan" untuk menscan *QR Code* yang sudah di enkripsi maupun belum dienkripsi. Pada pengujian waktu enkripsi dan dekripsi dilakukan dengan memasukkan sebuah *plaintext* dan key sesuai dengan *test vectors* AES. Lalu pengujian fungsional dan *non*-fungsional dilakukan dengan metode *black box testing* dengan mencoba seluruh fungsional yang ada pada sistem.

5.3.1 Pengujian *Test Vector*

Pengujian *test vectors* dimulai dengan memasukkan sebuah Key dan sebuah Plaintext yang didapat dari jurnal menurut Adam Barent (2014). Hasil yang didapat dari program kemudian akan di bandingkan dengan hasil dari proses manualisasi.

Pada tabel 5.1 merupakan tabel hasil dari pengujian test vectors dari proses enkripsi dengan menngunakan *key* "That's my Kung Fu" *plaintext "*Two One Nine Two". dan pada tabel 5.2 merupakan tebel hasil dari pengujian test vectors dari proses dekripsi dengan menngunakan *key* "That's my Kung Fu" *plaintext "*Two One Nine Two"

Tabel 5.1 Pengujian *test vectors* enkripsi

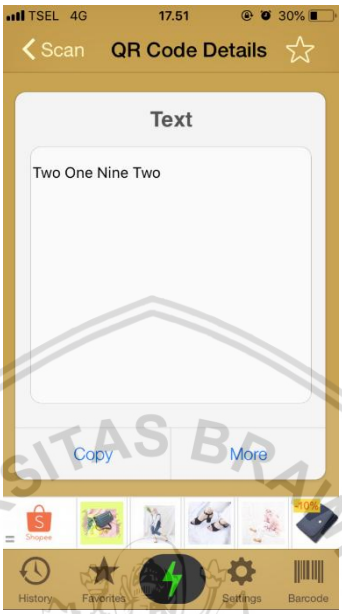
No	*Round*	Hasil dari program				Status
		Addroundkey	*Subtitusi byte*	*Shiftrow*	*Mix columns*	
1	0	00 3C 6E 47 1F 4E 22 74 0E 08 1B 31 54 59 0B 1A	-	-	-	Valid
2	1	58 15 59 CD 47 B6 D4 39 08 1C E2 DF 8B BA E8 CE	63 EB 9F A0 C0 2F 93 92 AB 30 AF C7 20 CB 2B A2	63 EB 9F A0 2F 93 92 C0 AF C7 AB 30 A2 20 CB 2B	BA 84 E8 1B 75 A4 8D 40 FA 8D 06 7D 7A 32 0E 5D	Valid
3	2	43 0E 09 3D C6 57 08 F8 A9 C0 EB 7F 62 C8 FE 37	6A 59 CB BD A0 4E 48 12 30 9C 98 9E 3D F4 9B 8B	6A 59 CB BD 4E 48 12 A0 98 9E 30 9C 8B 3D F4 9B	15 C9 7F 9D CE 4D 4B C2 89 71 BE 88 65 47 97 CD	Valid
4	3	78 70 99 4B 76 76 3C 39 30 7D 37 34 54 23 5B F1	1A AB 01 27 B4 5B 30 41 D3 BA 39 D2 AA E8 BB 9A	1A AB 01 27 5B 30 41 B4 E9 D2 D3 BA A9 AA E8 BB	AA 65 FA 88 16 0C 05 3A 3D C1 DE 2A B3 4B 5A 0A	Valid
5	4	B1 08 04 E7 CA FC B1 B2 51 54 C9 6C ED E1 D3 20	BC 51 EE B3 38 38 EB 12 04 FF 9A 18 20 26 39 A1	BC 51 EE B3 38 EB 12 38 9A 18 04 FF A1 20 26 39	10 BC D3 F3 D8 94 E0 E0 53 EA 9E 25 24 40 73 7B	Valid
6	5	9B 23 5D 2F 51 5F 1C 38 20 22 BD 91 68 F0 32 56	C8 30 F2 94 74 B0 C8 37 D1 20 DD 50 55 F8 66 B7	C8 30 F2 94 B0 C8 37 74 DD 50 D1 20 B7 55 F8 66	2A 26 8F E9 78 1E 0C 7A 1B A7 6F 0A 5B 62 00 3F	Valid
7	6	14 8F C0 5E 93 A4 60 0F 25 2B 24 92 77 EB 40 75	14 26 4C 15 D1 CF 9C 07 B7 93 7A 81 45 8C 23 B1	14 26 4C 15 CF 9C 07 D1 7A 81 B7 93 B1 45 8C 23	A9 37 AA F2 AE D8 0C 21 E7 6C B1 9C F0 FD 67 3B	Valid
8	7	53 43 4F 85 39 06 0A 52 8E 93 3B 57 5D F8 95 BD	F4 73 BA 58 DC 49 D0 76 3F F1 36 4F F3 9B 09 9D	F4 73 BA 58 49 D0 76 DC 36 4F 3F F1 9D F3 9B 09	9F 37 51 37 AF EC 8C FA 63 39 04 66 4B FB B1 D7	Valid
9	8	66 70 AF A3 25 CE D3 73 3C 5A 0F 13 74 A8 0A 54	ED 1A 84 97 12 6F 67 00 19 DC E2 5B 4C 41 2A 7A	ED 1A 84 97 6F 67 00 12 E2 5B 19 DC 7A 4C 41 2A	E8 8A 4B F5 7475 EE E6 D3 1F 75 58 55 8A 0C 38	Valid
10	9	09 A2 F0 7B 66 D1 FC 3B 8B 9A E6 30 78 65 C4 89	33 51 79 0A 3F 8B 66 8F EB BE 76 7D 92 C2 67 20	33 51 79 0A 8B 66 8F 3F 76 7D EB BE 20 92 C2 67	B6 E7 51 8C 84 88 98 CA 34 60 66 FB E8 D7 70 51	Valid
11	10	29 57 40 1A C3 14 22 02 50 20 99 D7 5F F6 B3 3A	01 3A 8C 21 33 3E B0 E2 3D B8 8E 04 BC 4D 1C A7	01 3A 8C 21 3E B0 E2 33 8E 04 3D B8 A7 BC 4D 1C	-	Valid

Tabel 5.2 Pengujian *test vectors* dekripsi

No	Round	Hasil dari program				Status
		Addroundkey	*Invers* Subtitusi byte	*Invers* Shiftrow	*Invers* Mix columns	
1	10	01 3a 8c 21 3e b0 e2 33 8e 04 3d b8 a7 bc 4d 1c	-	-	-	Valid
2	9	b6 e7 51 8c 84 88 98 ca 34 60 66 fb e8 d7 70 51	09 a2 f0 7b d1 fc 3b 66 e6 30 8b 9a 89 78 65c4	09 a2 f0 7b 66 d1 fc 3b 8b 9a e6 30 78 65 c4 89	33 51 79 0a 8b 66 8f 3f 76 7d eb be 20 92 c2 67	Valid
3	8	e8 8a 4b f5 74 75 ee e6 d3 1f 75 58 55 8a 0c 38	66 70baf a3 ce d3 73 25 0f 13 3c 5a 54 74 a8 0a	66 70 af a3 25 ce d3 73 3c 5a 0f 13 74 a8 0a 54	ed 1a 84 97 6f 67 00 12 e2 5b 19 dc 7a 4c 41 2a	Valid
4	7	9f 37 51 37 af ec 8c fa 63 39 04 66 4b fb b1 d7	53 43 4f 85 06 0a 52 39 3b 57 8e 93 bd 5d f8 95	53 43 4f 85 39 06 0a 52 8e 93 3b 57 5d f8 95 bd	fa 73 ba 58 49 d0 76 dc 36 4f 3f f1 9d f5 9b 09	Valid
5	6	a9 37 aa f2 ae d8 0c 21 e7 6c b1 9c f0 fd 67 3b	14 8f c0 5e a4 60 0f 93 24 92 25 2b 75 77 e8 40	14 8f c0 5e 93 a4 60 0f 25 2b 24 92 77 e8 40 75	14 26 4c 15 cf 9c 07 d1 7a 81 b7 93 b1 45 8c 23	Valid
6	5	2a 26 8f e9 78 1e 0c 7a 1b a7 6f 0a 5b 62 00 3f	9b 23 5d 2f 5f 1c 38 51 bd 91 20 22 56 68 f0 32	9b 23 5d 2f 51 5f 1c 38 20 22 bd 91 68 f0 32 56	c8 30 f2 94 b0 c8 37 74 dd 50 d1 20 b7 55 f8 66	Valid
7	4	10 bc d3 f3 d8 94 e0 e0 53 ea 9e 25 24 40 73 7b	b1 08 04 e7 fc b1 b2 ca c9 6c 51 54 20 ed e1 d3	b1 08 04 e7 ca fc b1 b2 51 54 c9 6c ed e1 d3 20	bc 51 ee b3 38 eb 12 38 9a 18 04 ff a1 20 26 39	Valid
8	3	aa 65 fa 88 16 0c 05 3a 3d c1 de 2a b3 4b 5a 0a	78 70 99 4b 76 3c 39 76 37 34 30 7d f1 54 23 5b	78 70 99 4b 76 76 3c 39 30 7d 37 34 54 23 5b f1	1a ab 01 27 5b 30 41 b4 e9 d2 d3 ba 9a aa e8 bb	Valid
9	2	15 c9 7f 9d ce 4d 4b c2 89 71 be 88 65 47 97 cd	43 0e 09 3d 57 08 f8 c6 eb 7f a9 c0 37 62 c8 fe	43 0e 09 3d c6 57 08 f8 a9 c0 eb 7f 62 c8 fe 37	6a 59 cb bd 4e 48 12 a0 98 9e 30 9c 8b 3d f4 9b	Valid
10	1	ba 84 e8 1b 75 a4 8d 40 f4 8d 06 7d 7a 32 0e 5d	58 15 59 cd b6 d4 39 47 e2 df 08 1c ce 8b ba e8	58 15 59 cd 47 b6 d4 39 08 1c e2 df 8b ba e8 ce	63 eb 9f a0 2f 93 92 c0 af c7 ab 30 a2 20 cb 2b	Valid
11	0	54 4f 4e 20 77 6e 69 54 6f 65 6e 77 20 20 65 6f	fb e9 db e0 15 dc 4f ba 79 c6 62 04 3a b7 1f f1	00 3c 6e 47 1f 4e 22 74 0e 08 1b 31 54 59 0b 1a	-	Valid

5.3.2 Pengujian Keamanan

Pada pengujian keamanan dilakukan dengan menggunakan aplikasi yang bernama “Quick Scan” untuk menscan *QR Code* yang sudah di enkripsi maupun belum dienkripsi.



Gambar 5.3 Tampilan kode QR *plaintext*

Gambar 5.4 Tampilan kode QR *ciphertext*

Pada gambar 5.3 merupakan QR yang berisi pesan asli, dan pada gambar 5.4 adalah QR yang telah di enkripsi isi nya sehingga orang lain tidak mengetahui isi pesan aslinya.

5.3.3 Pengujian Waktu Enkripsi dan Dekripsi

Pada pengujian waktu enkripsi dan dekripsi dilakukan dengan memasukkan sebuah *plaintext* dan *key* sesuai dengan *test vectors* AES. Hasil dari seriap *round* akan dimasukkan kedalam tabel 5.3 dan 5.4. Tujuan dari pengujian waktu enkripsi dan dekripsi adalah untuk mengetahui waktu tiap *rounds* dan jumlah waktu keseluruhan dimulai dari round 0-10 pada AES 128/128 bit.

Tabel 5.3 Tabel pengujian waktu enkripsi

Round ke-	Nama *Round*	Waktu
0	*Round key*	1.4066696166992E-5 DETIK
1	*SubBytes*	1.9073486328125E-5 DETIK
	ShiftRows	1.0967254638672E-5 DETIK
	MixColumns	2.9802322387695E-5 DETIK
	AddRoundKey	3.6001205444336E-5 DETIK
2	*SubBytes*	3.7193298339844E-5 DETIK
	ShiftRows	1.6927719116211E-5 DETIK
	MixColumns	4.2915344238281E-5 DETIK
	AddRoundKey	2.9087066650391E-5 DETIK
3	*SubBytes*	3.6001205444336E-5 DETIK
	ShiftRows	2.1934509277344E-5 DETIK
	MixColumns	6.5088272094727E-5 DETIK
	AddRoundKey	4.6968460083008E-5 DETIK
4	*SubBytes*	5.0067901611328E-5 DETIK
	ShiftRows	2.3126602172852E-5 DETIK
	MixColumns	7.8201293945312E-5 DETIK
	AddRoundKey	8.9883804321289E-5 DETIK
5	*SubBytes*	8.2969665527344E-5 DETIK
	ShiftRows	4.0054321289062E-5 DETIK
	MixColumns	4.3153762817383E-5 DETIK
	AddRoundKey	0.00017118453979492 DETIK
6	*SubBytes*	2.598762512207E-5 DETIK
	ShiftRows	2.0027160644531E-5 DETIK
	MixColumns	4.1007995605469E-5 DETIK
	AddRoundKey	2.7179718017578E-5 DETIK
7	*SubBytes*	2.1934509277344E-5 DETIK
	ShiftRows	1.3828277587891E-5 DETIK
	MixColumns	3.6001205444336E-5 DETIK
	AddRoundKey	3.1948089599609E-5 DETIK
8	*SubBytes*	2.3841857910156E-5 DETIK
	ShiftRows	1.5020370483398E-5 DETIK
	MixColumns	3.504753112793E-5 DETIK
	AddRoundKey	2.7894973754883E-5 DETIK
9	*SubBytes*	3.4809112548828E-5 DETIK
	ShiftRows	1.5974044799805E-5 DETIK
	MixColumns	3.6001205444336E-5 DETIK
	AddRoundKey	2.8848648071289E-5 DETIK
10	*SubBytes*	2.0980834960938E-5 DETIK
	ShiftRows	1.4066696166992E-5 DETIK
	AddRoundKey	1.0967254638672E-5 DETIK
Total waktu yang dibutuhkan		0.0034630298614502 DETIK

Pada tabel 5.3 merupakan tabel hasil dari pengujian waktu enkripsi 128 bit *key* dan 128 bit *plaintext* dan terlihat waktu enkripsi yang kurang lebih sama besar pada setiap *rounds.*

Tabel 5.4 Tabel pengujian waktu dekripsi

Round ke-	Nama *Round*	Waktu
10	*Round key*	1.215934753418E-5 DETIK
9	*InvSubBytes*	4.0054321289062E-5 DETIK
	InvShiftRows	2.7894973754883E-5 DETIK
	InvMixColumns	5.0067901611328E-5 DETIK
	AddRoundKey	1.3113021850586E-5 DETIK
8	*InvSubBytes*	2.3126602172852E-5 DETIK
	InvShiftRows	1.4066696166992E-5 DETIK
	InvMixColumns	4.6014785766602E-5 DETIK
	AddRoundKey	1.0967254638672E-5 DETIK
7	*InvSubBytes*	0.00037097930908203 DETIK
	InvShiftRows	1.5020370483398E-5 DETIK
	InvMixColumns	3.7193298339844E-5 DETIK
	AddRoundKey	1.3113021850586E-5 DETIK
6	*InvSubBytes*	2.1934509277344E-5 DETIK
	InvShiftRows	1.4066696166992E-5 DETIK
	InvMixColumns	4.4107437133789E-5 DETIK
	AddRoundKey	1.0013580322266E-5 DETIK
5	*InvSubBytes*	2.4080276489258E-5 DETIK
	InvShiftRows	1.5020370483398E-5 DETIK
	InvMixColumns	3.504753112793E-5 DETIK
	AddRoundKey	1.0967254638672E-5 DETIK
4	*InvSubBytes*	2.3126602172852E-5 DETIK
	InvShiftRows	1.7166137695312E-5 DETIK
	InvMixColumns	3.504753112793E-5 DETIK
	AddRoundKey	1.0967254638672E-5 DETIK
3	*InvSubBytes*	2.288818359375E-5 DETIK
	InvShiftRows	1.2874603271484E-5 DETIK
	InvMixColumns	3.814697265625E-5 DETIK
	AddRoundKey	1.0967254638672E-5 DETIK
2	*InvSubBytes*	2.288818359375E-5 DETIK
	InvShiftRows	1.3113021850586E-5 DETIK
	InvMixColumns	3.6954879760742E-5 DETIK
	AddRoundKey	1.0967254638672E-5 DETIK
1	*InvSubBytes*	2.3126602172852E-5 DETIK
	InvShiftRows	1.5974044799805E-5 DETIK
	InvMixColumns	4.6968460083008E-5 DETIK
	AddRoundKey	1.1920928955078E-5 DETIK
0	*InvSubBytes*	3.3855438232422E-5 DETIK
	InvShiftRows	1.6927719116211E-5 DETIK
	AddRoundKey	1.215934753418E-5 DETIK
Total waktu yang dibutuhkan		0.0029640197753906 DETIK

Pada tabel 5.4 merupakan tabel hasil dari pengujian waktu dekripsi 128 bit *key* dan 128 bit *plaintext* dan terlihat waktu dekripsi yang kurang lebih sama besar pada setiap *rounds.*

5.3.4 Pengujian Fungsional

Tabel 5.5 Pengujian fungsional

No	*Test Name*	*Test Case*	*Expected Result*	Result	Status
1	Pengujian Memasukkan *Key*	*user* mengimputkan *key*	Sistem merespon dengan menampilkan inputan Key di laman enkripsi/ dekripsi.	Sistem merespon dengan menampilkan inputan Key di laman enkripsi/ dekripsi.	valid
2	Pengujian Memasukkan *plaintext*	*user* mengimputkan *plaintext*	Sistem merespon dengan menampilkan inputan *plaintext* di laman enkripsi.	Sistem merespon dengan menampilkan inputan *plaintext* di laman enkripsi.	valid
3	Pengujian Melakukan Enkripsi AES	*User* menekan tombol *"Encrypt"*	Sistem merespon dengan menampilkan proses enkripsi dari round 1-10.	Sistem merespon dengan menampilkan proses enkripsi dari round 1-10.	valid
4	Pengujian Merubah *plaintext* Menjadi *QR Code*	*User* menekan tombol "*QR Code*"	Sistem merespon dengan menampilkan *QR Code*	Sistem merespon dengan menampilkan *QR Code*	Valid
5	Pengujian Melakukan Scan *QR Code*	*User* menekan tombol "nyalakan"	Sistem merspon dengan menampilkan hasil Scan.	Sistem merspon dengan menampilkan hasil Scan.	Valid
6	Pengujian Melakukan Dekripsi AES	*User* menunjukan gambar *QR Code* ke kamera perangkat yang sedang digunakan.	Sistem merespon dengan menampilkan proses dekripsi dari state awal sampai akhir.	Sistem merespon dengan menampilkan proses dekripsi dari state awal sampai akhir.	Valid

Pada tabel 5.3 merupakan tabel hasil dari pengujian fungsional yang telah dilakukan pada sistem AES.

5.3.5 Pengujian *Non*-Fungsional

Tabel 5.6 Pengujian *non*-fungsional

No	*Test Name*	*Test Case*	*Expected Result*	Result	Status
1	*Portability*	Sistem dijalankan di browser Google Chrome, Mozilla Firefox, dan Opera pada sistem operasi Windows.	AES berjalan tanpa masalah dan tampilan tidak berantakan	AES berjalan tanpa masalah dan tampilan tidak berantakan	Valid

Pada tabel 5.4 merupakan tabel hasil dari pengujian non-fungsional yang telah dilakukan pada sistem AES.

BAB 6 PENUTUP

Bagian ini memuat kesimpulan dan saran terhadap skripsi. Kesimpulan dan saran disajikan secara terpisah, dengan penjelasan sebagai berikut:

6.1 Kesimpulan

Sesuai dengan hasil implementasi algoritme AES untuk dekripsi dan enkripsi menggunakan *QR Code*, maka dapat ditarik kesimpulan sebagai berikut :

1. Algoritme AES dapat diterapkan pada QR-*Code.* Algoritme AES akan memberikan aspek *confidentiality,* hal ini dapat dibuktikan dengan pengujian keamanan. Algoritme AES pada proses enkripsi menghasilkan output berupa *ciphertext* berupa karakter tidak jelas yang sulit dipahami dan pada proses dekripsi dilakukan dengan menscan *ciphertext* QR-*Code* dan memasukkan *key* lalu sistem akan menjalankan proses dekripsi AES dan menghasilkan isi pesan asli atau *plaintext.*
2. Berdasarkan pengujian validasi *plaintext dan ciphertext* pada algoritme AES dapat dibuktikan pada pengujian *test vector* pada algoritme AES. Pada setiap hasil *output* program dari setiap *round* proses enkripsi maupun dekripsi setelah dibandingkan menunjukan hasil yang valid secara keseluruan.
3. Berdasarkan hasil pengujian kinerja pemrosesan enkripsi dan dekripsi algoritme AES pada QR *Code* membutuhkan waktu enkripsi 0.0034 detik dan dekripsi membutuhkan waktu 0.0029 detik.

6.2 Saran

Pada penelitian ini terdapat beberapa saran yang dapat digunakan untuk penelitian selanjutnya. Pertama, Objek *QR Code* bisa diganti dengan objek lain misalnya barcode atau gambar. Kedua, AES ini dapat digantikan dengan algoritme enkripsi dan dekripsi lainnya agar dapat mengetahui algoritme mana yang lebih baik untuk objek tertentu. Ketiga, algoritme kedua atau ke tiga dapat ditambahkan untuk meningkatkan keamanan dari hasil enkripsi dan dekripsi.

DAFTAR PUSTAKA

Ariadi, 2011. *Analisis Dan Perancangan Kode Matriks Dua Dimensi Quick Response (QR) Code.*. [online] Repository.usu.ac.id. Available at: <http://repository.usu.ac.id/handle/123456789/29816> [Accessed 30 June 2018].

Ariyus, D., 2006. *Kriptografi Keamanan Data Dan Komunikasi*. Yogyakarta: Graha Ilmu.

Barent, A., 2014. [online] Adamberent.com. Available at: <http://www.adamberent.com/documents/AESbyExample.pdf> [Accessed 2 July 2018].

Daemen, J. and Rijmen, V., 1999. The Rjndael Block Cipher. AES Proposal : Rijndael. *The Rjndael Block Cipher*, [online] Available at: <https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf> [Accessed 30 June 2018].

denso, A., 2011. [online] Nacs.org. Available at: <http://www.nacs.org/LinkClick.aspx%3Ffileticket%3DD1FpVAvvJuo%253D%26tabid%3D1426%26mid%3D4802> [Accessed 30 June 2018].

Denso, 2013. [online] Denso Wave. Available at: <http://www.qrcode.com/> [Accessed 17 June 2018].

Harahap, M., 2016. Analisis Perbandingan Algoritma Kriptografi Klasik Vigenere Cipher dan One Time Pad. InfoTekJar (Jurnal Nasional Informatika dan Teknologi Jaringan), [online] 1(1), pp.61-64. Available at: <https://jurnal.uisu.ac.id/index.php/infotekjar/article/view/43/30> [Accessed 18 July 2018].

Haryana, K., 2018. *Pengembangan Perangkat Lunak Dengan Menggunakan PHP*. [online] Jurnal.stmik-mi.ac.id. Available at: <http://jurnal.stmik-mi.ac.id/index.php/jcb/article/view/74> [Accessed 30 June 2018].

Kavaliro.com. 2014. [online] Available at: <https://kavaliro.com/wp-content/uploads/2014/03/AES.pdf> [Accessed 2 July 2018].

Kurniawan, Y., 2007. Perbandingan Analisis Sandi Linear Terhadap AES, DES, dan AE1. *JUTI: Jurnal Ilmiah Teknologi Informasi*, 6(2), p.57.

Munir, R., 2006. *Kriptografi*. 1st ed. Bandung: Informatika Bandung.

Sholeh, M. and Muharom, L., 2016. Smart Presensi Menggunakan *QR Code* Dengan Enkripsi Vigenere Cipher. Limits: Journal of Mathematics and Its Applications, 13(2), p.31.

Sugiyono, 2010. *Metode Penelitian Pendidikan Pendekatan Kuantitatif, Kualitatif, Dan R D*. Bandung: Alfabeta.

Sumardi, 2017. Studi Model Algoritma Kriptografi Klasik dan Modern. [online] Available at: <http://seminar.uny.ac.id/semnasmatematika/sites/seminar.uny.ac.id.semnasmatematika/files/full/T-37.pdf> [Accessed 17 July 2018].

Widiyanto, A., 2007. *Meningkatkan Keamanan Komputer Anda*. 2nd ed. semarang: NEOMEDIA PRESS.Ariadi, 2011. Analisis dan Perancangan Kode Matriks Dua Dimensi Quick Response (QR) code. *Skripsi Universitas Sumatera Utara.*